Abstract:

Systems and methods for automated monitoring and management of distributed applications, client/server databases, networks and systems across heterogeneous environments. Distributed, automated intelligent monitoring agents use embedded sensing technology which is knowledgeable of application protocols, to monitor continuously the network environment in real time. To this end, the monitoring agent can be located on each client and server in the network. The monitoring agent can couple to the communications stack for monitoring the data that is being passed between the client and the network, of a server in the network. The data can be collected and employed for trouble shooting trend analysis, resource planning, security auditing, and accounting as well as other applications. Also included is a controller for remotely coordinating the data gathering process from the various clients and servers. Data gathering can be performed in accordance with trigger events or on a periodic basis. Data may also be associated with a transaction and gathered in accordance with business transaction rules.

특2001-0079612

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.[7]
G06F 15/00

(11) 공개번호 특2001-0079612
(43) 공개일자 2001년08월22일

(21) 출원번호 10-2001-7001497
(22) 출원일자 2001년02월03일
번역문제출일자 2001년02월03일
(86) 국제출원번호 PCT/US1999/17531
(87) 국제공개번호 WO 2000/08806
(86) 국제출원출원일자 1999년08월03일
(87) 국제공개일자 2000년02월17일
(81) 지정국 국내특허 : 아랍에미리트 알바니아 아르메니아 오스트리아 오스트레일리아 아제르바이잔 보스니아-헤르체고비나 바베이도스 불가리아 브라질 벨라루스 캐나다 스위스 중국 쿠바 체코 독일 덴마크 에스토니아 스페인 핀랜드 영국 그레나다 그루지야 가나 감비아 크로아티아 헝가리 인도네시아 이스라엘 인도 아이슬란드 일본 케냐 키르기즈 북한 대한민국 카자흐스탄 세인트루시아 스리랑카 라이베리아 레소토 리투아니아 룩셈부르크 라트비아 몰도바 마다가스카르 마케도니아 몽고 말라위 멕시코 노르웨이 뉴질랜드 폴란드 포르투칼 루마니아 러시아 수단 스웨덴 싱가포르 슬로베니아 슬로바키아 시에라리온 타지키스탄 투르크메니스탄 터어키 트리니다드토바고 우크라이나 우간다 우즈베키스탄 베트남 유고슬라비아 남아프리카 짐바브웨 AP ARIPO특허 : 가나 감비아 케냐 레소토 말라위 수단 시에라리온 스와질랜드 우간다 짐바브웨

EA 유라시아특허 : 아르메니아 아제르바이잔 벨라루스 키르기즈 카자흐스탄 몰도바 러시아 타지키스탄 투르크메니스탄

EP 유럽특허 : 오스트리아 벨기에 스위스 사이프러스 독일 덴마크 스페인 핀랜드 프랑스 영국 그리스 아일랜드 이탈리아 룩셈부르크 모나코 네덜란드 포르투칼 스웨덴

OA OAPI특허 : 부르키나파소 베닝 중앙아프리카 콩고 코트디브와르 카메룬 가봉 기네 기네비쏘 말리 모리타니 니제르 세네갈 차드 토고

(30) 우선권주장 60/095,142 1998년08월03일 미국(US)
60/137,121 1999년06월02일 미국(US)

(71) 출원인 콩코드 커뮤니케이션즈 인코포레이티드 더글라스 에이. 베트
미합중국 매사추세츠주 01752 말보로 넉커슨 로드 600

(72) 발명자 윌슨제임스
미합중국메사추세츠02492,니드햄,싸우스스트리트35
에게르월네에라즈
미합중국뉴햄프셔03110,베드포드,호크드라이브88
페르난데즈게리
미합중국메사추세츠01742,콩코드,로즈브룩로드358
닥터머타자
미합중국메사추세츠01801,앤도버,펀플라이트써클1
케인켄
미합중국메사추세츠01720,엑톤,오버룩드라이브25
브리너알비트
미합중국뉴햄프셔03049,홀리스,트위스레인64
머다나쉐카
미합중국메사추세츠02451,윌스엄,스테언즈힐로드5708
디그루트피터
미합중국메사추세츠01827,던스테이블,피.오.박스306

라이온-스미스존
미합중국메사추세츠01886,웨스포드,케일라드라이브10
멘델스코트
미합중국메사추세츠01854,로웰,콘스탄스드라이브84
(74) 대리인 김학제, 문혜정

심사청구 : 없음

(54) 진단 정보를 이용한 분산 어플리케이션들의 모니터링시스템 및 방법

요약

본 발명은 이종 환경에 분산되어 있는 분산 어플리케이션, 클라이언트/서버 데이터베이스, 네트웍 및 시스템들을 자동 모니터링하고 관리하는 시스템 및 방법이다. 분산된, 자동 지능 모니터링 대리자들은 어플리케이션 프로토콜이 알 수 있는 내장 감지 기술(embedded sensing technology)을 이용하여 실시간으로 네트웍 환경을 지속적으로 모니터한다. 이러한 목적을 위해, 모니터링 대리자는 네트웍에서 각 클라이언트 및 서버에 존재할 수 있다. 모니터링 대리자는 네트웍의 서버의 네트웍의 클라이언트와 서버 사이에서 전달되는 데이터를 모니터하는 통신 스택에 커플링할 수 있다. 데이터는 수집되고 장애 추적 경향 분석(trouble seeting trend analysis), 자원 계획, 보안 검사, 및 과금은 물론 다른 어플리케이션들을 위해 이용될 수 있다. 또한, 본 발명에는 여러 클라이언트들 및 서버들로부터의 데이터 수집 프로세스를 원격적으로 조정하는 컨트롤러가 포함된다. 데이터 수집은 트리거 이벤트에 따라 수행되거나 정기적으로 이루어질 수 있다. 데이터는 트랜스액션과 관련되고 비즈니스 트랜스액션 규칙에 따라 수집된다.

대표도

도1

색인어

분산 응용, 클라이언트, 서버, 컨트롤러, 대리자, 콘솔, 네트웍, 모니터링

명세서

[관련 출원에 대한 참조]

본 출원은 1998년 8월 3일자 미합중국 가득허출원 제 60/095,142호, 및 1999년 6월 2일자 미합중국 가득허출원 제 60/137,121호를 우선권 주장의 기초로 하고, 현재 출원중인 1997년 3월 20일자 미합중국특허출원 제 08/821,698호의 일부계속출원이다.

기술분야

본 발명은 일반적으로 분산 컴퓨터 환경(distributed computing environment)을 모니터하고 관리하는 시스템 및 방법에 관한 것으로, 더욱 상세하게는 분산 컴퓨터 시스템(distributed computing system)의 기업 광역 오퍼레이션(enterprise wide operation)을 모니터하는 시스템 및 방법에 관계한다.

배경기술

분산 컴퓨터 아키텍처(distributed computing architecture)는 네트웍 시스템에 의해 서로 접속된 다수의 컴퓨터들 사이의 물리적이고 논리적인 분산 컴퓨팅 기능(distribution computing functions)을 제공한다. 전형적으로, 클라이언트는 네트웍을 통해서 서버에게 일정한 서비스 요청을 보낸다. 서버는 하나 이상의 데이터베이스, 파일, 프린팅 또는 기타의 서비스를 수행함으로써 클라이언트의 요청에 응답한다. 이러한 오퍼레이션 중에, 클라이언트와 서버는 데이터를 서로 교환하고 오퍼레이션을 완료하는데 필요한 데이터 처리 기능을 개별적으로 수행한다. 하나의 서버가 복수의 클라이언트들에 대해 동시에 서비스할 수 있어야 하고, 동시에 클라이언트가 다수의 서버들의 서비스들에 동시에 엑세스할 수 있도록 하려면 복잡해진다. 더욱이, 서버들은 다른 서버들에 대해서는 클라이언트로 기능할 수도 있다. 따라서, 분산 컴퓨터 시스템은 복잡하고, 많은 다층 구조의 분산 아키텍처(distributed architecture)를 가질 수 있다.

복잡성에도 불구하고, 분산 컴퓨터 아키텍처는 많은 양의 데이터를 효과적으로 처리하고 다수의 스테이션들 사이의 신속한 디지털 통신을 제공하는 복잡하고 막강한 시스템을 사용자들에게 성공적으로 제공해 왔다. 이러한 시스템들의 파워는 분산 컴퓨터 아키텍처의 광범위한 확산을 초래하였고 클라이언트/서버 데이터베이스, 분산 어플리케이션(distributed application) 및 상이한 환경 사이의 네트웍과 같은 과다한 분산 컴퓨터 서비스의 개발을 초래하였다. 더욱이, 새로운 기술의 개발로 분산 시스템의 성장은 가속화되었다. 예를 들어, 상업 환경에 적합한 인터넷 및 인트라넷 시스템의 개발은 분산 컴퓨터 분야의 폭발적인 성장을 가져왔다.

분산 컴퓨터 아키텍처가 사용자들에게 효율적이고 막강한 수단을 제공한다고 해도, 구조의 복잡성과 정교함은 실제 시스템의 구현, 분해 및 작동을 어렵게 만든다. 예를 들어, 전형적인 관계 클라이언트/서버 데이터베이스 시스템은 복수의 클라이언트들에 대해 다수의 데이터베이스 서비스를 제공하기 위한 데이터베이스 서버를 포함할 것이다. 분산 아키텍처는 일반적으로 각 클라이언트가 서버와 적절하게 통신할 수 있는 능력을 보유하고, 서버는 각 클라이언트들로부터 수신한 많은 서비스 요청을 조정하고, 몇몇 네트웍 메모리 장치들 사이에 분산되어 존재할 수 있는 데이터 저장소에 대해 데이터의 일관성(data coherency)를 유지할 것을 요구한다. 클라이언트 구성요소들과 서버들 사이에 일어나는 통신은 비동기적으로, 간헐적으로 매우 빠르게 일어나기 때문에, 이러한 시스템을 컴퓨터 네트웍에 로딩하는 것은 어렵고 복잡한 일이다. 따라서, 복잡한 진단 및 관리 수단이 이들 분산 시스템을 구현하고 분석하고 성능을 개선하기 위해 이용될 수 있다.

분산 컴퓨터 아키텍처의 복잡성은 진단 시스템(diagnosing system)이 제대로 작동하지 않게 만들어 성능 평가를 어렵게 만든다. 분산된 네트웍 구성요소들 사이의 통신의 비동기 및 고속 특성은 성능 평가를 매우 어렵게 만든다. 따라서, 진단 기술자는 시스템의 오작동 또는 성능 병목 현상과 같은 성능 문제를 일으키는 원인을 탐지하기 위해 시스템 작동을 모니터할 때 어려움을 겪을 수 있다.

이러한 진단 및 개발 수단의 요구에 부응하여, 컴퓨터 엔지니어들은 네트웍의 통신 채널에 결합되어 클라이언트와 서버 사이의 트랜스액션들을 모니터하는 네트웍 모니터 시스템을 개발하였다. 이들 시스템들은 자주 네트웍 시스템의 물리 계층에 삽입되어 통신을 모니터하는 하드웨어 장치이다. 따라서, 이것은 클라이언트와 서버 사이의 각각의 물리적인 접속이 서로 접속된 하드웨어 장치(interconnected hardware device)를 포함할 것을 필요로 한다. 이러한 장치들은 일어나는 데이터 트랜스액션들을 모니터한다. 이러한 데이터 트랜스액션의 기록을 생성함으로써, 시스템 기술자들은 시스템 오작동 및 성능 열화를 초래하는 이벤트를 파악하려고 시도할 수 있다.

이러한 시스템들이 작동하더라도, 하드웨어 장치들이 클라이언트와 서버 사이에 일어나는 각각의 데이터 트랜스액션을 검출하고 기록할 수 있어야 한다. 이것은 하드웨어 장치가 네트웍을 통해서 전달되는 각 데이터 패킷을 읽어서 전송되는 데이터가 모니터 대상이 되는 클라이언트 또는 서버와 관련이 있는지 판단할 것을 요구한다. 그러나, 클라이언트와 서버 사이에 일어나는 데이터 트랜스액션의 비동기 및 고속 특성 때문에, 이러한 장치들은 모든 트랜스액션을 검출하지 못하는 오류를 범하기 쉽다. 기술자들은 클라이언트와 서버 사이에 일어나는 트랜스액션들의 일부 기록, 즉, 시스템 오작동 및 성능 문제의 원인을 규명하기 위한 목적으로 사용하기에 불완전한 기록만을 가질 수 있다.

대리자-콘솔 아키텍처(agent-console architecture)를 구현함으로써 중앙 시스템 관리 모델과 분산 환경을 매핑시키는 다른 관리 툴(management tools)이 존재한다. 이러한 아키텍처에서, 대리자는 지속적으로 시스템의 서버 및 로그파일들, 시스템, 네트웍 또는 어플리케이션을 폴링하여 사용 데이터(usage data)를 수집하고 '예외(exception)'가 발생했는지 조사한다. 콘솔은 중앙 관리 스테이션으로 이것을 통해 명령과 제어 기능이 구현된다. 이러한 아키텍처는 몇 가지 단점을 갖는다. 첫 째, 지속적인 폴링 기능은 귀중한 자원을 사용하여 서버의 성능을 저하시킨다. 이것은 특히 시스템 활동의 세밀하고 정밀한 분석을 필요로 하고 지속적인 폴링을 요구하는 메트릭(metrics)에서 특히 그러하다. 둘 째, 대리자는 서버 구성요소 레벨(server component level)이다. 따라서, 사용, 성능, 및 예외 통계(exception statistics)는 구성요소 레벨에서만 이용가능하고 종단간(end-to-end) 자원 활용에 대해서는 평가가 이루어지지 않고 다른 관련된 구성요소들에 대해서도 평가가 이루어지지 않는다. 또한, 데이터 수집 기능은 실시간으로 수행되지 않을 수 있다.

특정 프레임워크 판매자들에 의해 제안된 다른 접근방법은 관리 수단이 시스템 성능을 모니터하기 위해 이용할 수 있는 한 세트의 자원(resources)에 대한 어플리케이션 프로그램 인터페이스(API)를 포함한다. 이러한 접근방법은 편집되고 재컴파일되는 시스템에서 운영되는 기존의 분산 어플리케이션들이 여러 가지 시스템 모니터 자원에 대한 API 요청을 포함할 것을 요구한다. 따라서, 이것은 일반적으로 분산 시스템에서 실행되는 어플리케이션 프로그램을 제공하는 모든 판매업자들의 협력에 의존하는 시스템 모니터링에 대해 크게 방해가 되는 접근방법이다.

[발명의 요약]

본 발명의 하나의 양상은 분산 컴퓨터 시스템의 모니터링 방법이다. 트리거 이벤트들과 수집될 관련 데이터가 정의된다. 클라이언트와 제 1 서버 사이의 접속을 모니터하면서 클라이언트에서 트리거 이벤트들중 하나의 발생을 검출한다. 클라이언트 데이터는 클라이언트에서 하나의 트리거 이벤트에 따라서 수집된다. 하나의 트리거 이벤트의 발생의 검출은 컨트롤러에 통보된다. 제 1 서버에게 트리거 이벤트의 발생을 통보한다. 제 1 서버 데이터는 제 1 서버에 의해 수집되고, 제 1 서버 데이터는 컨트롤러로 전송된다.

본 발명의 다른 양상은 분산 컴퓨터 시스템의 모니터링 시스템이다. 기계 실행 코드(machine executable code)는 트리거 이벤트 및 수집될 관련 데이터를 정의한다. 기계 실행 코드는 클라이언트와 제 1 서버 사이의 접속을 모니터하면서 클라이언트에서의 트리거 이벤트들중 하나의 발생을 검출한다. 기계 실행 코드는 클라이언트에서 하나의 트리거 이벤트에 따라 클라이언트 데이터를 수집한다. 기계 실행 코드는 컨트롤러에게 하나의 트리거 이벤트의 검출을 알린다. 기계 실행 코드는 제 1 서버에게 트리거 이벤트의 발생을 통보한다. 기계 실행 코드는 제 1 서버에 의해 제 1 서버 데이터를 수집하고 기계 실행 코드는 제 1 서버 데이터를 컨트롤러에 전송한다.

도면의 간단한 설명

본 발명의 특징 및 이점들은 첨부 도면과 이하의 바람직한 실시예의 상세한 설명으로부터 더욱

자명해질 것이다.

도 1은 분산 어플리케이션 (distributed applications)들의 자동 모니터 및 관리 기능을 제공하는 본 발명의 시스템의 일실시예의 예시도이다.

도 2는 도 1에 도시된 시스템에서의 실행에 적합한 모니터링 대리자(monitoring agent)의 블럭도의 일실시예의 예시도이다.

도 3은 분산 프로세스의 세부사항들을 수집하는 모듈의 일실시예의 예시도이다.

도 4는 모듈의 동작 세부사항의 일실시예의 더욱 상세한 예시도이다.

도 5는 분산 프로세스의 세부사항들을 수집하는 모듈의 다른 실시예의 예시도이다.

도 6은 분산 어플리케이션들을 모니터하는데 이용될 수 있는 시스템의 일실시예의 예시도이다.

도 7은 성능 데이터를 수집하는 방법의 각 단계들의 일실시예의 예시도이다.

도 8은 클라이언트에서 정의된 이벤트에 따라 데이터를 수집하는 세부 단계들의 일실시예의 예시도이다.

도 9는 컨트롤러가 클라이언트 데이터를 받고 서버 접속을 이루는 방법을 결정하기 위한 세부 단계들의 일실시예의 예시도이다.

도 10은 서버 데이터가 수집되는 방법의 세부 단계들의 방법의 일실시예의 예시도이다.

도 11은 도 6의 시스템의 데이터 저장소에 포함될 수 있는 테이블의 일실시예의 예시도이다.

도 12는 캐시 데이터 값들의 테이블의 일실시예의 예시도이다.

도 13은 비즈니스 트랜스액션 시스템의 일실시예의 예시도이다.

도 14는 트랜스액션 엔진의 기본적인 오퍼레이션의 방법 단계들을 설명한 흐름도의 일실시예의 예시도이다.

도 15는 데이터가 비즈니스 트랜스액션 명세에 적합한지 여부를 결정하는 세부 단계들을 도시한 흐름도의 일실시예의 예시도이다.

도 16은 비즈니스 트랜스액션 시스템과 대리자 사이의 관계의 개략도의 일실시예의 예시도; 및

도 17은 비즈니스 트랜스액션 명세(business transaction specification)를 보존하는 시스템 구성요소들의 블럭도의 일실시예의 예시도이다.

발명의 상세한 설명

도 1은 복수의 워크스테이션 및 서버들을 포함하는 분산 어플리케이션 (distributed application)을 모니터하고 관리하는 본 발명의 시스템(10)의 구현예이다. 시스템(10)은 도면에 도시하지 않은 프린터 및 네트웍 메모리 장치와 같은 네트웍 장치들을 추가로 포함할 수 있다는 것을 이해할 수 있을 것이다. 특히, 도 1은 워크스테이션들(12, 14, 16 및 18), 서버(20), 서버(22), Mission Universal Monitor(MUM) 대리자(30-40), MUM 콘솔 모듈(42), MUM 데이터베이스(44), 및 모니터링 스테이션(24)를 도시하고 있다.

도 1에 도시된 시스템 10에서, 각각의 워크스테이션들(12-18)은 컴퓨터가 판독할 수 있는 정보를 갖는데 적합한 네트웍에 접속된 컴퓨터 시스템이다. 각각의 네트웍 노드들은 서버 20 또는 22중 하나와 통신할 수 있고, 그들로부터 서비스를 요청할 수 있다. 따라서, 도 1에 도시된 시스템은 서로 다른 워크스테이션들, 장치들, 및 서버들 사이에 분포되어 있는 컴퓨팅 기능(computing functions)의 물리적 및 논리적 분포를 나타내는 분산 컴퓨터 아키텍처(distributed computing architecture)를 갖는다.

도 1은 이러한 분산 처리 환경을 모니터하는 본 발명의 시스템의 구조와 동작의 개관을 보여 준다. 시스템 10은 MUM 대리자 30-40와 MUM 데이터베이스 44와 접속되어 있는 MUM 콘솔 42를 포함한다. 각각의 MUM 대리자들 30-40은 각각 하나의 워크스테이션 또는 서버 구성요소와 접속되어 있다. 더욱이, 모니터링 대리자(monitoring agent)는 각각의 네트웍 구성요소와 접속되어, 분산된 것을 포함하는 모든 프로세스와 같이, 클라이언트에 기업 광역 모니터링(enterprise wide monitoring)을 제공하고 서버들을 모니터할 수 있다. 각각의 MUM 대리자는 물리적으로 그의 관련된 워크스테이션에 또는 서버에 존재하여, 특히 워크스테이션에서 동작하는 프로그램과 선택된 하나 이상의 서버들 20 또는 22 사이에 교환되는 데이터를 모니터한다.

결론적으로, 네트웍상의 모든 장치의 프로세싱이 모니터될 수 있고 각각의 도시된 대리자들 30-40은 MUM 콘솔 모듈 42에 접속되어 수집된 데이터들을 나타내는 정보를 MUM 콘솔 42로 보낸다. MUM 콘솔 42는 이러한 정보를 운영자(예컨대, 시스템 기술자 또는 시스템 관리자)에 의해 분석되도록 중앙 데이터베이스 44에 저장한다. 그렇지 않으면, 대리자들은 직접 MUM 데이터베이스에 정보를 제공할 수 있다. 콘솔 42 또는 기타의 다른 시스템에서 실행되는 어플리케이션 프로그램은 수집된 데이터를 검토해 보고 구성요소 레벨 및 기업 레벨에서 기업의 임의의 프로세스 또는 구성요소의 시스템 성능을 보여줄 수 있다. 더욱이, 시스템 관리자는 기업 레벨 사용 통계(enterprise level usage statistics) 및 반응 시

간, 개발 챠트 및 보고서를 작성할 수 있고 기업의 오퍼레이션과 관련된 사용자-지정 통계를 알아보기 위한 임의의 다른 관련 데이터 분석을 수행할 수 있다.

따라서, MUM 대리자 30-40의 기능은 데이터를 수집하는 것이다. 이러한 목적을 위해, 각각의 도시된 MUM 대리자 30-40은 대리자를 갖는 모든 노드들이 관리 노드(managed nodes)로 인식되는 기업의 관리 노드에 위치되는 지능 모니터링 대리자(intelligent monitoring agents)일 수 있다. 도 1에 도시된 시스템 10에서, 기업체의 각각의 노드들은 각각의 MUM 대리자들을 갖는데, 컴퓨터 분야의 통상의 지식을 가진 당업자들에게 기업 내에 분포되어 있는 MUM 대리자들의 수가 다양하고, 어플리케이션의 특성에 따라 선택될 수 있다는 것이 자명할 것이다. MUM 대리자들은 도시된 워크스테이션 및 서버에서 실행되어 본 발명에 따른 시스템에서 서버, 워크스테이션 및 네트웍을 적합하게 구성하는 C** 생성 컴퓨터 프로그램과 같은 소프트웨어 모듈일 수 있다. 그러나, 당업자들은 MUM 대리자들이 개개의 관리 노드들의 백플레인에 접속되어 로컬 시스템의 동작을 모니터하는 전자회로 카드 어셈블리와 같은 하드웨어 장치이거나 또는 하드웨어와 소프트웨어가 조합된 것일 수도 있다는 것을 이해할 것이다. 이러한 임의의 구현예들은 본 발명의 정신을 벗어나지 않는 범위 내에서 실시될 수 있다.

각각의 MUM 대리자들 30-40은 자동적으로 동작하여 로컬 프로세서 성능, 로컬 프로세서 자원, 로컬 프로세서 구조, 분산 어플리케이션 등의 오퍼레이션, 네트웍의 오퍼레이션, 디스크, 파일 시스템 장치 및 테이프를 포함하는 다양한 네트웍 장치들의 오퍼레이션 및 기타의 정보들을 포함하는 다수의 구성요소들을 모니터한다. 따라서, 본 발명의 구현은 진단 분석(diagnostic analysis)이 서버 CPU 성능의 측정 이상을 이용할 수 있다는 것이다. 각각의 모니터링 대리자 30-40은 기업 모니터 콘솔 42에 의한 일정한 폴링을 피하는 정보를 포함할 수 있는, 단일의, 다-스레드 프로세스(multi-threaded process)일 수 있다. 각각의 대리자들은 비즈니스 트랜스액션, 데이터베이스, 시스템 및 네트웍 검출 및 관련 이벤트들, 수정 액션(corrective action)의 개시, 및 이벤트 통보의 제공을 지속적으로 실시간으로 모니터할 수 있다. MUM 대리자들은 비즈니스 트랜스액션을 이해할 수 있고 이벤트들, 자원 사용, 및 반응시간의 세부사항을 수집할 수 있고 그러한 정보를 MUM 콘솔 42로 보내어 그것을 MUM 데이터베이스 44에 저장할 수 있다.

각각의 대리자들 30-40도 루프 백 오퍼레이션(loop back operation)을 모니터 할 수도 있다. 도시된 바와 같이, 루프 백 23, 25는 서버와 분산 어플리케이션들이 동일한 클라이언트에 위치될 때 일어난다. 루프 백 정보는 네트웍으로 나가지 않고, 통신 스택(communication stack)을 통해서 서버와 어플리케이션 사이에 전달된다. 각 대리자 38, 40이 클라이언트 통신 스택에 접속되어 있기 때문에, 그것은 통신 스택을 통해서만 전달되는 루프 백 데이터를 모니터할 수 있다.

도 2는 도 1에 도시된 시스템 10에 이용하기에 적합한 MUM 대리자 50의 아키텍처 기능블럭도로 도시한 것이다. 대리자 50은 관리 노드상에서 백그라운드 프로세스와 같이 실행되는 소프트웨어 모듈일 수 있다. 특히, 대리자 50은 SYBASE™ 클라이언트로 기능할 수 있는 네트웍 노드상에 존재하는 대리자일 수 있다. 대리자 50은 외부 이벤트 인터페이스 52, 통신 인터페이스 54, 툴 인터페이스 58, 및 MUM 콘솔 인터페이스 및 이벤트 상관 프로세서(event correlation processor) 64, 시스템 모니터 70, 네트웍 모니터 72, SYBASE™ 클라이언트 모니터 74, 및 SYBASE™ 서버 모니터 76을 포함한다.

대리자 50의 전체적인 아키텍처는 대리자가 한 세트의 모니터 구성요소들 70-76과 기업의 다양한 구성요소들에 관한 이벤트 정보를 상관 프로세서 64(correlation processor)에 제공하는 외부 이벤트 인터페이스 52를 포함한다는 것을 보여준다. 상관 프로세서 64는 이벤트들을 서로 관련시켜 MUM 콘솔 42로 전달되거나 또는 다른 관리 툴 또는 알람을 설정하거나, 호출기를 활성화하거나, 모뎀을 통해서 팩스를 보내거나, 시스템 관리자에게 이메일을 보내거나 수정 액션(corrective action)을 취하기 위한 기계 사용 코드(instrumentation code)를 포함하는 다른 툴로 전달될 수 있는 데이터를 생성한다. 따라서, 대리자 50은 이벤트들의 세부사항을 수집하고 이러한 세부사항을 상관 프로세서 64에서 처리하여 다른 여러 가지 중에서 비즈니스 트랜스액션을 나타내는 정보를 생성한다.

이러한 목적을 위해, 외부 이벤트 인터페이스 52와 모니터 구성요소들 70-76은 이벤트들에 관한 세부사항을 수집할 수 있다. 외부 이벤트 인터페이스 52는 특정 이벤트를 찾아내기 위해 로컬 시스템에서 실행되는 하나의 세트의 프로그램 모듈일 수 있다. 이들 외부 인터페이스들은 여러 장치들로부터 사용자 및 시스템 프로세스로부터의 이벤트를 추적하는 Simple Network Management Protocol(SNMP) 일 수 있다.

모니터 구성요소들 70-76은 특정 이벤트에 관한 정보를 수집하기 위해 노드에서 실행되는 코드 모듈(code module)일 수 있고, 이러한 코드 모듈들로부터 요청을 받아 검출된 이벤트를 모니터 구성요소들에게 통보하는 프로그래밍 인터페이스(programming interface)를 포함할 수 있다. 모니터 구성요소들 70-76은 대리자 50의 경우, 시스템 이벤트, 네트웍 이벤트, 및 SYBASE 클라이언트/서버 이벤트들을 포함할 수 있는, 로컬 시스템에 관련된 임의의 기업 구성요소들을 모니터하는 코드로부터 이벤트 통보를 받을 수 있다. 모니터 구성요소들 70-76의 프로그래밍 인터페이스는 특정 이벤트의 검출시 이러한 코드로부터 요청을 받을 수 있는 익스포트 C** 기초 API일 수 있다. 예를 들어, 노드는 일반 보호 폴트(general protection fault)를 트래핑하는 실행 코드(running code)일 수 있는데, 이러한 일반 보호 장애는 시스템

이벤트로 정의될 수 있다. 코드는 시스템 모니터 구성요소 70에게 검출된 폴트를 알리기 위해 대리자 50에게 API 호를 발하고 시스템 모니터 구성요소 70은 통보를 상관 프로세서 64에게 보낼 수 있다. API는 다른 대리자 구성요소의 오퍼레이션으로부터 코드의 오퍼레이션 특성을 제거하기 위해 코드의 서비스 프로토콜을 캡슐화할 수 있다. 이것은 검출 코드가 사용자가 어느 이벤트가 모니터되어야 하는지 선택할 수 있는 플러그-인 모듈(plug-in module)로 기능하게 만든다. 더욱이, 사용자는 대리자에 의한 모니터링을 위해 목적으로 하는 이벤트들을 정의할 수 있다. 대리자 50내에 있는 임의의 모니터 구성요소들의 이벤트 세부사항을 전달하는데 적합한 어떠한 코드라도 본 발명의 정신을 벗어나지 않고 본 발명에 따라 실시될 수 있다.

특히, 시스템 모니터 70은 로컬 시스템의 오퍼레이션에 관한 정보를 수집할 수 있다. 이러한 목적을 위해, 시스템 모니터 70은 프로세서 부하, 메모리 사용, 이용가능한 메모리 스페이스에 관한 정보 및 로컬 워크스테이션 또는 서버의 오퍼레이션을 설명해 줄 수 있는 기타의 유사한 정보를 수집하는 코드 모듈(code module)을 포함할 수 있다. 이러한 코드의 개발은 본 발명이 속하는 기술분야에 잘 알려져 있고, 임의의 적합한 코드가 본 발명의 정신을 벗어나지 않는 범위 내에서 본 발명에 의해 실시될 수 있다. 또한, 각각의 대리자는 로컬 시스템에 대해서 알람 설정, 호출기 활성화, 모뎀을 통한 팩스 전송, 시스템 관리자에 대한 이메일 발송 또는 수정 액션(corrective action))을 취하는 것과 같은 액션을 취할 수 있다.

시스템 모니터 70과 대조적으로, 모니터 72-76은 분산 프로세스와 관련된 이벤트들을 모니터 하는 것이다. 따라서, 모니터된 이벤트들과 관련된 세부사항들은 진행되고 있는 분산 프로세싱 오퍼레이션(distributed processing operations)에 관한 정보를 필요로 한다. 본 발명의 구현은 모니터 72-76이 분산 프로세스 동안에 일어나는 네트웍 통신을 모니터함으로써 분산 프로세스에 관한 정보를 수동적으로 수집하는 것이다. 이러한 목적을 위해, 각 모니터 요소는 분산 프로세스 구성요소들 사이의 통신을 수동적으로 모니터하기 위해서 네트웍 통신 스택에 대한 인터페이스를 포함할 수 있다.

도 3은 분산 프로세스의 세부사항들을 수집하는, SUN SOLARIS ™ 플랫트폼에서 이용하기에 적합한 모듈 80의 일실시예를 도시한 것이다. 모듈 80은 로컬 클라이언트 쪽에 있는 데이터베이스 소프트웨어 51과 선택된 서버 그룹들 사이의 네트웍 통신을 모니터하여, 통신 내용의 카피를 대리자 50에게 보내어 처리하고 있다면 어떤 이벤트가 일어났는지 판단하게 한다. 특히, 모듈 80은 로컬 클라이언트의 데이터베이스 소프트웨어 51(예컨대, SYBASE, CORBA, ORACLE)의 대리자 50 또는 관리 노드의 TCP 스택을 통해서 전달하는 트래픽을 갖는 다른 분산 프로세스 어플리케이션 53에 의한 모니터링을 가능하게 할 수 있다.

모듈 80은 대리자 50 및 STREAMS 모듈 55로 구성되어 있다. 대리자 50은 TAP API 57과 소켓 라이브러리 59를 포함한다. TAP 모듈 61와 TCP STREAMS 드라이버 63을 포함하는 STREAMS 모듈 55는 더블유. 리처드 스티븐스의 교재 Unix Network Programming에 설명된 것을 포함하여, 컴퓨터 공학분야에서 잘 알려진 원리에 따라 구성될 수 있다. STREAMS 모듈 55는, 도 3에 도시된 바와 같이, 커널 모드(kernel mode)로 실행될 수 있다. TAP 모듈 61은 관련 노드 안팎의 모든 TCP 트래픽이 모듈 80을 통과하도록 TCP STREAMS 스택 65의 정상에 오토푸시된다. 모듈 80은 ORACLE 또는 SYBASE와 같은 모니터될 분산 프로세스 또는 프로세스들의 통신 프로토콜을 알고 있다. 이러한 방식으로, 모듈 80은 트래픽으로부터 대리자 50에 관련된 부분들을 걸러내어 이러한 트래픽의 카피를 대리자 50에게 전달할 수 있다.

도 4는 SUN SOLARIS ™ 플랫트폼상의 모듈 80(도 3에 도시된)의 오퍼레이션 세부사항들의 상세한 도면이다. 모듈 80은 모니터될 서버들의 리스트 84를 보존하고 있다. 하나의 구현예에서, 서버들의 리스트는 대리자 50에 의해 모듈 80에 제공되고, 대리자는 그 리스트를 MUM 콘솔 42로부터 받을 수 있다. 리스트된 서버들에 접속되는 모든 인바운드 및 아웃바운드 트래픽은 모듈 80에 의해 대리자 50에 전달된다.

동작시, 각각의 접속된 스트림(connected stream)에 대해서, 모듈 80은 정보를 저장하기 위한 컨텍스트 데이터 구조(context data structure)를 만든다. 이어서 컨텍스트는 도시된 바와 같이, 대응하는 STREAMS 대기행렬(queue)에 접속되어 읽고 쓴다. 컨텍스트는 접속의 타입을 설명하는 접속된 스트림에 관한 정보를 저장한다. 모듈 80은 대리자 50과 통신하기 위해 컨텍스트 데이터 구조 94를 향상한다. 모듈 80은 컨텍스트 94내에 이벤트 세부사항들에 관한 데이터를 저장할 수 있고, 대리자 50은 데이터를 독출하여 정보를 상관 프로세서 64에 전달할 수 있다. 탭 테이블(tap table) 82는 접속된 스트림들 컨텍스트 데이터 구조들의 리스트, 모니터될 모든 서버들의 리스트, 및 대리자 50과 통신하는 컨텍스트 데이터 구조를 저장한다.

각 시간에 접속은 컨텍스트를 만드는 모듈을 확립한다. 이어서 모듈은 아래로 보내지는 M_IOCTL 요청의 시퀀스를 생성함으로써 접속을 위한 서버 주소를 결정한다. TCP 스택으로부터의 응답이 포착되고 상류로 보내지지 않도록 차단되며, 서버 IP 주소가 분석된다. 만약 주소가 리스트 84에 있는 주소들중 어느 하나와 매치되면 접속은 모니터 대상으로 표시되고, 모듈 80은 모니터링을 시작한다. 모니터링 중에, 모니터된 접속의 컨텍스트 하에서 송수신된 모든 트래픽의 카피들은 상류로 대리자 50에게 전달되기 위해, 대리자 통신 컨텍스트 94의 상류 대기행렬에 전달된다. 이러한 방식으로 대리자 50은

임의의 선택된 서버들에 대해 접속하기 위한 모든 트래픽의 카피들을 수신한다.

모듈 80에 의해 상류로 대리자 50에 전달된 정보는 이벤트 상관 프로세서 58로 보내진다. 이벤트 상관 프로세서는 이벤트 상세 정보를 처리하여 특정한 트랜스액션 레벨 정보를 알아내는 소프트웨어 모듈이다. 예를 들어, 모듈 80은 SYBASE 로그온 요청을 대리자 50에 전달할 수 있다. 이벤트 상관 프로세서는 로그온 요청을 받아 SYBASE 서버에 의해 로그온 확인이 올 때까지 접속을 모니터한다. 이러한 두 가지 이벤트들 사이의 시간차를 비교함으로써, 이벤트 상관 프로세서 58은 분산 프로세스의 반응시간(response time)의 크기를 측정할 수 있다. 생성될 수 있는 다른 유사한 메트릭들도 생성되어 시스템 성능의 종단간 레벨 분석(end-to-end level analysis)을 제공할 수 있다.

도 5는 분산 프로세스의 세부사항들을 수집하는, WINDOW NT™ 플래트폼에서 사용하기에 적합한 모듈의 다른 예(180)를 도시한 것이다. 모듈 180은 모듈 80과는 다른 아키텍처를 이용하지만, 동일한 기능을 수행한다. 즉, 모듈 180은 로컬 클라이언트의 데이터베이스 소프트웨어 151 또는 시스템 서비스 176에 의해 관리 노드의 TCP 스택을 통해 전달되는 기타의 분산 프로세스 어플리케이션들 153, 155의 대리자 150에 의한 모니터링을 가능하게 한다.

모듈 180은 대리자 150 및 TAP 드라이버 163을 포함한다. 대리자 150은 TAP API 157을 포함하고, TAP 드라이버 163은 Tap 장치 161과 TapFilter 장치 162를 포함한다. Tap 장치 161은 TAP API 157와 통신하여 모니터링 파라미터를 설정하고 모니터링 데이터를 제공한다. TapFilter 장치 162는 TCP/IP 드라이버 165의 TCP 장치 167의 정상에 부설되어 관리노드 안팎의 모든 TCP 트래픽들을 모니터 한다. TapFilter 장치 162는 WINDOW NT™ 드라이버 레이어링 아키텍처(driver layering architecture)의 특징들을 이용하여 자신을 TCP 장치 167에 부착시킨다. 특히, TapFilter 장치 162는 운영 체제 호(즉, IoAttachDevice)를 이용하여 그 자신을 TCP 장치에 대한 데이터 스트림내에 삽입시킨다. 따라서, TCP 장치내외로 전달되는 모든 데이터는 TapFilter 장치 162를 통해서 전달된다. 시스템 서비스 176, Tap 드라이버 163, 및 TCP/IP 드라이버는 도 5에 도시된 바와 같이, 커널 모드로 동작할 수 있다.

TCP/IP 드라이버 165도 UDP 장치 169와 TCP/IP군의 다른 프로토콜중의 다른 서브-프로토콜을 서포트하는 IP 장치 171을 포함한다. 도시하지는 않았으나, TapFilter 장치 162는 UDP 장치 169 및/또는 IP 장치 171의 위에 부설되어 이러한 장치들의 모든 데이터 트래픽을 모니터할 수 있다.

도시된 모듈 80과 180은 네트웍 통신 스택과 인터페이스 되는 모듈의 실시예들을 예시한 것뿐이다. 다른 모듈들도 본 발명에 따라 실시될 수 있다. 예를 들어, 도 1을 다시 참조하면, 시스템 10이 이종의 오퍼레이션 환경들을 갖는 분산 시스템이라는 것을 보여주기 위해서 서로 다른 타입의 워크스테이션들이 시스템 10에 도시되어 있는 것을 확인할 수 있다. 워크스테이션들은 상이한 아키텍처, 하드웨어 및 소프트웨어일 수 있고, 서버들 20 및 22는 마찬가지로 상이한 하드웨어 또는 소프트웨어 아키텍처일 수 있다. 예를 들어, 워크스테이션 12는 도 2에 도시된 바와 같은 모듈을 갖는 SUN 워크스테이션일 수 있다. 더욱이, 워크스테이션 및 서버들과 서로 접속되는 네트웍 시스템은 근거리통신망, 광역망, 대도시망, 또는 이들을 조합한 것일 수 있다. 포인트는 분산 컴퓨터 아키텍처에서 특정 서비스를 구현하기 위해 일어나는 프로세싱은 그것이 이메일, 데이터베이스, 워드프로세싱 프로그램, 또는 기타 다른 서비스 또는 컴퓨터 어플리케이션이든간에, 데이터를 교환하기 위해 어느 정도 서로 접속된 다수의 프로세서 사이에 분포되어 있다는 것이다.

데이터 수집 활동을 설명한다. 이전 문단은 대리자를 설명한 것인데, 이러한 설명은 서버와 클라이언트에도 해당된다. 후술하는 문단에서는, 분산 컴퓨터 시스템을 모니터하고 데이터를 수집하는 프로세스를 설명한다.

도 6을 참조하면, 도 6은 분산 어플리케이션 들을 모니터 하는데 이용될 수 있는 시스템의 일실시예를 도시한 것이다. 도 6의 구현예는 시스템 210과 클라이언트 212a-212n 및 서버 214a-214n 포함한다. 또한, 클라이언트 212a-212n 및 서버 214a-214n와 상호작용하는 컨트롤러 216는 도 6의 다양한 클라이언트 및 서버 시스템 상에서 실행되는 분산 어플리케이션들을 모니터한다. 컨트롤러 216은 데이터 저장소 220에 읽고 쓴다. 마찬가지로, 콘솔 218은 데이터 저장소 220으로부터 데이터를 읽고 쓸 수 있다. 일반적으로, 컨트롤러 216은, 후술하는 바와 같이, 도 6의 시스템에서 실행되는 모니터링 어플리케이션들에 대한 데이터 수집 프로세스에 대한 드라이버이다. 데이터는 데이터 저장소 220으로부터 읽히고 기록된다. 콘솔 218은 인터페이스로 기능하는데, 예컨대, 사용자가 새로운 보고와 같은 데이터 저장소에 저장된 데이터의 해석을 읽기를 원하는 경우 또는 데이터 저장소 220에 저장된 다양한 파라미터들을 변경하고자 하는 경우에 인터페이스로 기능한다.

도 6의 시스템은 도 1의 시스템 10과 관련하여 앞서 설명한 것과 유사한 구성요소들을 포함하고 있다는 것을 유의해야 한다. 예를 들어, 클라이언트 212a-212n는 도 1의 시스템에 도시된 워크스테이션 또는 PC일 수 있다. 마찬가지로, 서버 214a-214n는 도 1에 도시된 SYBASE SQL 서버와 같은, 데이터베이스 서버중의 하나일 수 있다.

각각의 클라이언트 212a-212n 및 각각의 서버 214a-214n 및 컨트롤러 216은 예를 들어, 전용 하드웨어 프로세서이기 보다 개념적인 기능 블록을 나타낸다. 한편, 본 발명의 원리에 따른 하나의 구현에는 하나의 시스템상의 복수의 클라이언트들 또는 컨트롤러는 물론 하나의 시스템상의 하나의 컨트롤러와 하나 이상의 서버들을 구비할 수 있다. 하나의 구현에에서, 컨트롤러 216은 컨트롤러가 제공하는 트래픽 및 관리 기능의 양 때문에 전용 프로세서에서 실행될 수 있다. 전용 프로세서 파워는 데이터 수집 및 관리 프로세스를 제어하고 조정하기 위해 요구된다. 다른 구현에에서, 클라이언트 또는 서버 및 컨트롤러와 관련된 기능을 특정 하드웨어에 할당하는 것은 네트웍 트래픽 및 클라이언트-서버 트랜스액션의 양은 물론, 시스템 210내의 상이한 하드웨어들의 다양한 처리속도에 따라 달라질 수 있다.

도 6의 구현예에서, 각각의 클라이언트와 서버들은 대리자와 연관된다. 예를 들어, 클라이언트 1(212a)와 관련된 것은 대리자 1(215a)이다. 일반적으로, 각각의 대리자들은 클라이언트 또는 관련된 서버 어플리케이션들을 수용하는 컴퓨터 시스템 상에서 실행되는 프로세스이다. 대리자는 소프트웨어이거나 또는 후술하는 바와 같은 예외 조건(exception condition)을 검출하는 프로세스이다. 마찬가지로, 컨트롤러 216은 대리자들중 하나와 콘솔 218 사이에 정보를 전달하는 코디네이터인 프로세스 또는 프로그램이다. 컨트롤러 216은 데이터 저장소 220에 정보를 저장하고 그곳의 정보에 엑세스한다.

도 6의 시스템 210의 다른 구현에는 각 서버에 대해 대리자를 포함하지 않을 수도 있다는 것을 유의해야 한다. 즉, 다른 바람직한 구현에에서, 대리자는 각 구현에에 따라 요구되는 경우 각 서버에 선택적으로 포함될 수 있다. 대리자가 각 서버에 포함될 지 아니면 서버와 관련된 선택적인 구성요소인지에 대한 결정은 각 구현예의 데이터 수집 요구에 따른다.

이러한 구현에에서 데이터 저장소 220는 도 6의 시스템 210의 클라이언트-서버 어플리케이션 활동과 관련된 구조(configuration) 및 사용 정보와 같은 다양한 유형의 데이터들을 보존하고 저장하기 위해 만들어진 데이터베이스이다. 데이터 저장소 220은 트리거 이벤트 데이터 및 다음 단락에서 더 자세하게 설명될 조건 및 임계값을 포함할 수도 있다. 또한, 하나의 구현에에서, 데이터 저장소 220은 지속 데이터 저장 장소(persistent data storage area)이다. 일반적으로, 이러한 구현예에서 데이터 저장소 220에 저장된 데이터는 시스템 210의 분산 컴퓨터 환경내의 다양한 클라이언트-서버 접속 사이의 모니터링 활동으로부터 수득된 모니터링 데이터를 포함한다. 데이터는 일반적으로 클라이언트와 서버 사이의 논리적인 통신 매체를 참조하여 접속을 모니터하는 것으로부터 구할 수 있다. 모니터되는 것은 모니터될 다양한 클라이언트와 서버 사이와 같은 접속 중에 전송되는 요청-응답 트래픽이다. 예를 들어, 클라이언트 1과 서버 n 사이에 접속이 있다면, 접속이 모니터되고 클라이언트 1과 서버 n 사이에 일어나는 네트웍 트래픽에 관련된 이벤트로 대응하는 데이터가 수집된다. 이러한 데이터는 컨트롤러에 의해 수집되고 특수한 비즈니스 트랜스액션 또는 접속에 관련된 콘솔 218 활동을 사용자에게 보고하는 것과 같은 다양한 용도를 위해 데이터 저장소 220에 저장된다. 더욱 상세한 실시예 및 설명은 관련 도면과 함께 후술한다.

후술할 것은 분산 데이터 컬렉션과 미리 정해진 이벤트들 및 임계값 정보에 따른 예외 및 이벤트 트리거링이다.

도 7를 참조하면, 도 7은 도 6의 시스템 210에서 데이터 수집을 수행하는 방법의 흐름도이다. 제 230 단계에서, 트리거 이벤트 또는 예외 조건이 정의된다. 또한, 특수한 데이터는 다양한 트리거 이벤트를 식별하고 연관짓는다. 이러한 구현예에서, 특수한 트리거 이벤트와 관련된 데이터는 일반적으로 진단 정보 및 액션으로 설명될 수 있다. 진단 정보는 도 7의 처리 단계와 함께 후술하겠지만, 트리거 이벤트가 발생했을 때 수집된 데이터를 설명한다. 액션은 일반적으로 이벤트 트리거의 검출에 따라 어떤 단계가 수행되었는지를 설명한다. 액션은 예를 들어, 전자메일을 보내거나, 호출하거나 다른 멀티미디어 이벤트가 일어나는 것을 포함할 수 있다. 또한, 특수한 액션은 정해진 트리거 이벤트에 따라 일부 조건을 수정하거나 완화시키는 수정 액션(corrective action)일 수 있다. 수정 액션은 예를 들어 다른 컴퓨터 시스템과의 상호 작용을 포함할 수 있다.

수집된 데이터는 일반적으로 데이터의 수집 방법에 따라 두 개의 카테고리를 형성하는 것으로 설명될 수 있다. 클라이언트 및 서버로부터 수집된 데이터는 대리자 또는 비-대리자 소스로부터 유래된 것일 수 있다. 일반적으로, 대리자에 의해 수집된 데이터는 대리자 데이터 소스(agent data source)를 갖는다고 말해진다. 대리자에 의해 수집되지 않은 다른 모든 데이터는 비-대리자 데이터 소스를 갖는다고 한다. 대리자 및 비-대리자 데이터는 다음 처리 단계에서 트리거 이벤트를 특정하고 그와 연관될 수 있다.

제 232 단계에서, 도 7의 흐름도를 참조하면, 대기 루프(wait loop), 또는 이벤트 평가 루프가 도시되어 있는데, 제 230 단계에서 정의된 트리거 이벤트의 검출을 위한 테스트가 진행된다. 제 232 단계에서, 트리거 이벤트가 검출되지 않았다면 루프의 위쪽으로 다시 리턴하여 트리거 이벤트가 발생할 때까지 계속적인 모니터링을 수행한다. 이러한 모니터링은 당업자들에게 알려진 다양한 방법에 의해 상이한 실시예로 구현될 수 있다는 것을 유의해야 한다. 예를 들어, 하나의 구현예는 소정의 시간 간격으로

트리거 이벤트의 발생을 체크하는 기계 실행 코드 내에서 통화중 대기 루프(busy wait loop)를 실행함으로써 트리거 이벤트를 모니터할 수 있다. 다른 구현에는 비동기 이벤트 및 예외 조건 검출을 이용하는 것과 같이 오퍼레이션 시스템 기능(operation system functionality)을 기능적으로 이용함으로써 트리거 이벤트를 모니터하고 검출할 수 있다.

제 230 단계는 제 232 단계의 이벤트 평가 루프 이전의 다양한 시점에서 행해질 수 있다는 것을 유의하여야 한다. 즉, 트리거 이벤트의 정의는 제 232 단계 이벤트 평가 루프에 의해 수행되는, 트리거 이벤트의 검출과 루스하게 커플링될 수 있다고 할 수 있다. 트리거 이벤트는 별도의 작업에 의해 '오프-라인'으로 정의될 수도 있다. 예를 들어, 트리거 이벤트는 다른 컴퓨터 시스템상의 약간 앞선 시간에 정의될 수 있다. 트리거 이벤트들 및 관련 데이터는 나중에 갱신될 수 있다.

제 232 단계에서 일단 트리거 이벤트가 검출되면, 제 234 단계로 진행하여 클라이언트에서 정의된 이벤트에 따라 데이터 수집이 진행된다. 즉, 이러한 특수한 구현예에서, 클라이언트 1과 같은 클라이언트 상에 존재할 수 있는 대리자들은 다양한 트리거 이벤트들과 관련 데이터를 정하는 클라이언트-로컬 데이터 파일에 저장된 데이터에 엑세스한다. 이러한 데이터는 이러한 구현예에서 로컬 카피(local copy)로 각 클라이언트 및 서버들에 저장될 수 있다. 클라이언트상의 대리자는 제 232 단계에서 수행된 모니터링 활동을 실행하고 하나의 트리거 이벤트가 일어나기를 기다린다. 클라이언트의 대리자가 트리거 이벤트의 발생을 검출하면, 그것은 클라이언트 쪽에 있는 데이터 파일에 국부적으로 저장되어 있는 정의된 이벤트들(defined events)에 따라 데이터를 수집한다.

데이터 파일은 데이터 저장소에도 저장될 수 있고, 각각의 클라이언트 및 서버 쪽에 국부적으로 저장될 수 있다. 일반적으로, 이러한 데이터는 어떤 특정한 이벤트가 모니터되어야 하고 어러 클라이언트와 서버들 사이와 같은 연관된 접속을 설명한다. 각각의 트리거 이벤트에 대해, 각각의 클라이언트에 위치하는 대리자는 앞부분에서 도면과 함께 설명한 바와 같이 모니터링 시스템을 이용하여 다양한 클라이언트-서버 접속의 활동을 모니터한다.

하나의 구현예에서, 데이터 저장소에 저장되어 있는 데이터 파일의 변경 또는 갱신이 있다면, 컨트롤러는 클라이언트들 및 서버들에 의해 이용되는 데이터의 여러 가지 카피들을 일치시킨다는 것을 유의해야 한다. 즉, 데이터 파일이 변경되거나 갱신되면, 예를 들어, 오프-라인 편집 과정에 의해 새로운 트리거 이벤트가 부가되면, 클라이언트들 및 서버들에 의해 이용되는 바와 같은 데이터의 여러 가지 카피들을 데이터 저장소에 있는 카피와 일치시킨다. 컨트롤러는 이러한 갱신을 검출하여 각 클라이언트 또는 서버가 동일한 버전의 데이터 파일에 엑세스하게 만드는 역할을 담당한다.

대리자에 의해 모니터 될 이들 요청들 각각에 대해, 모니터 된 접속에 부착된 태스크(task)는 평가 컨텍스트 정보(evaluation context information)를 이용하여 평가된다. 평가 컨텍스트는 일반적으로 모니터될 접속에 관련된 정보를 포함한다. 평가 컨텍스트는 예를 들어, 사용자, 서버에 관한 정보 및 어플리케이션 요청(application request)과 관련된 다른 정보들과 함께 어플리케이션에 의해 실행될 명령을 포함한다. 트리거 이벤트 또는 예외 조건이 일어나면, 제 234 단계에서, 클라이언트 쪽에서 데이터 수집이 일어난다. 제 234 단계에서 수집된 데이터는 평가 컨텍스트 정보로 포함될 수 있다.
일반적으로, 클라이언트 쪽에서 대리자에 의해 이 시점에서 수집된 데이터의 유형은 클라이언트에 의해 계산될 수 있는 파라미터들 또는 변수들과 관계 있다. 이들은 전형적으로 사용자가 실행하는 시스템의 CPU 및 메모리 사용에 관한 자원 메트릭(resource metric)은 물론 클라이언트-서버 접속과 관련될 수 있는 대기시간과 같은 네트웍 파라미터와 같은 아이템들이다.

각각의 클라이언트 시스템에서 대리자를 실행시키는 것 이외에, 사용자 특정 프로그램(user specified program)이 추가적으로, 또는 그 대신에 실행될 수 있고 클라이언트 프로세스와 상호 작용하여 상태 정보, 예컨대, 도 6의 시스템 210에서 실행될 수 있는 다른 제품들 또는 프로세스들에 대한 정보를 수집할 수 있다.

제 236 단계에서, 평가 컨텍스트 정보를 포함하는 데이터는 클라이언트로부터 컨트롤러 216으로 보내진다. 제 238 단계에서, 컨트롤러 216은 클라이언트 데이터를 수신하여 방금 클라이언트 시스템에서 실행되는 대리자로부터 보고된 특정 모니터된 클라이언트-서버 접속의 서버 쪽에 접속한다. 예를 들어, 클라이언트 1과 서버 1 사이에 모니터되는 접속이 존재하고, 알람 조건 또는 트리거 이벤트가 클라이언트 1(212a)의 프로세스 내에서 실행되는 대리자 1(215a)에 의해 검출되었다면, 컨트롤러 216은 이러한 알람 조건의 발생을 통보 받는다. 결과적으로, 제 238 단계에서, 컨트롤러 216은 클라이언트 1(212a)로부터 보고된 데이터를 받고 서버 n(214n)에 접속한다. 제 240 단계에서, 특정한 모니터된 클라이언트-서버 접속의 서버로부터 서버 데이터가 수집된다. 이러한 접속 및 제어는 컨트롤러 216에 의해 시작된다. 이러한 능력으로, 컨트롤러 216은 클라이언트 1(212a)로 알려진 클라이언트들중 하나에 의해 트리거 이벤트 또는 예외조건의 발생을 통보받은 후에는, 데이터 수집 프로세스의 코디네이터 또는 드라이버로 기능한다.

제 238 단계에서, 컨트롤러는 클라이언트와 통신되는 다양한 조각의 정보들을 이용하여 어떤 서버와 접촉해야 되는지를 결정한다. 제 240 단계에서, 컨트롤러 216은 모니터되는 클라이언트-서버 접속의 서버로부터 서버 데이터를 수집한다. 이러한 데이터는, 예를 들어, 현재 열린 트랜스액션의 수 및 타입, 서버에 의해 서비스될 요청의 수, 및 클라이언트를 대신해서 앞서 수집한 것과 유사한 다양한 사용 통계를 포함할 수 있다. 클라이언트와 관련해서 앞서 설명한 것과 마찬가지로, 사용자-특정 프로그램 또는 프로세서는 컨트롤러 기계에서 실행될 수 있고, 컨트롤러 프로세서 216와 연관되어 도 6의 시스템 210에서 확인된 것과는 다른 제품들 및 소프트웨어에 의해 보존된 상태 정보를 수집할 수 있다는 것을 유의해야 한다.

제 240 단계에서, 서버 데이터는 모니터된 클라이언트-서버 접속의 서버로부터 수집된다. 서버로부터 수집된 데이터는 파일 또는 도 6의 시스템 210내의 서버들 및 클라이언트들 각각에 로컬 카피(local copy)가 존재하는 데이터 저장소 내에서 정의될 수 있는 변수 또는 파라미터들을 포함한다. 클라이언트와 관련해서 앞서 설명한 것과 마찬가지로, 접촉된 서버는 로컬 파일로부터의 데이터에 엑세스하여 그것이 어떤 변수 또는 파라미터를 계산할지를 결정하고 결과적으로 컨트롤러 216에 다시 전송한다. 제 242 단계에서, 컨트롤러는 부가적으로 모니터된 클라이언트-서버 접속에 포함된 임의의 다른 서버에 접속하고, 통보하고, 그로부터 데이터를 수신한다. 이러한 다른 서버들은, 예를 들어, 제 242 단계에서 모니터된 클라이언트-서버 접속에 관련된 트랜스액션을 보조하고 처리하는 서버들을 포함할 수 있다.

클라이언트 및 컨트롤러와 관련하여 앞서 설명한 바와 같이, 사용자-특정 프로그램 또는 프로세스는 상태 정보를 수집하기 위해 실행되거나 서버 프로세스와 결합될 수 있다. 상기 프로세스는, 예를 들어, 도 6의 시스템 210에는 도시되어 있지 않은 다른 소프트웨어일 수 있다.

도 8을 참조하면, 클라이언트에서 정의된 이벤트에 따른 데이터 수집의 하나의 구현에의 흐름도에서 더욱 상세한 단계들을 도시하였다. 제 234a 단계에서, 클라이언트는 특정 트리거 이벤트에 따라 정의된 클라이언트 상태 정보를 수집한다. 제 234b 단계에서 모니터된 클라이언트 서버 접속에 관련된 네트웍 상태 정보도 수집된다. 전술한 바와 같이, 이것은 모니터 클라이언트-서버 접속을 처리함에 있어서의 네트웍 대기지연 시간과 같은 정보를 포함한다.

도 9는 컨트롤러가 어떻게 클라이언트 데이터를 수신하고 서버 또는 서버들과 접속을 이루는지를 판단하는 과정의 흐름도의 세부 단계를 도시한 것이다. 제 250 단계에서, 컨트롤러는 클라이언트 컨텍스트 정보를 이용하여 데이터 저장소 220내의 다른 정보에 엑세스한다. 이러한 클라이언트 컨텍스트 정보는 그 대리자가 클라이언트 서버 접속을 모니터하는 클라이언트 시스템중 하나에 의해 전송된 예외 식별자(exception identifier)와 같은 데이터를 포함할 수 있다. 이러한 식별자를 이용해서, 컨트롤러는 데이터 저장소에 저장된 정보에 인덱싱하여, 제 252 단계에서, 어떤 정보가 클라이언트로부터 수집될지를 결정한다. 컨트롤러는 데이터 저장소를 이용하여, 예컨대, 모니터링시 또는 데이터 수집 과정에서 도 6의 시스템의 어떤 서버 또는 다른 프로세스들과 접촉할 필요가 있는지를 결정한다.

도 10은 도 7의 제 240 단계와 관련하여 전술한 바와 같이 모니터된 클라이언트-서버 접속의 서버로부터 서버 데이터를 수집하는 과정의 상세 흐름도이다. 제 240a 단계에서, 서버 상태 정보가 수집된다. 서버 상태 정보는, 예컨대, 앞서 설명한 바와 같이, 오픈 트랜스액션의 수 및 현재 특정 서버에 의해 서비스되고 있는 요청의 수와 같은 서버로부터의 비-대리자 데이터(non-agent data)를 포함할 수 있다. 일반적으로, 이러한 비-대리자 데이터 및 다른 서버 상태 정보는 214n과 같은 특정 서버에 의해 실행되고 연관되는 대리자 프로세스에 의해 수집된다. 제 240b 단계에서, 서버는 특정 서버에 저장된 이벤트 데이터 파일의 로컬 카피 내에 정해진 것들에 따라 서버 상에서 계산되어야만 하는 임의의 변수들 또는 파라미터들을 계산한다. 제 240c 단계에서, 컨트롤러는 모니터되는 클라이언트-서버 접속의 클라이언트에 의해 최초에 이루어지거나 검출되는 트리거 이벤트의 발생을 유발한 요청(request)과 관련된 서버로부터 상태 정보를 수집한다.

도 11을 참조하면, 도 11은 데이터 저장소 220에 저장되는, 이러한 구현에에서 각각의 클라이언트 및 서버들에 대해 국부적으로 여러 가지 클라이언트 트리거 이벤트들 또는 예외들과 관련된 조건을 한정하는 테이블의 하나의 구현에의 예시도이다. 일반적으로, 이러한 테이블은 전술한 도 7의 제 230 단계에서 도입된 구조 데이터(configuration data)의 일례인 데이터를 포함한다. 이러한 테이블은, 예컨대, 도 7과 함께 앞서서 설명했던 여러 가지 처리 단계들에서 어떤 변수가 각각의 클라이언트들 및 서버들에 의해 수집되어야 하는지를 결정하는데 이용될 수 있다. 이러한 구현에의 테이블 260은 모니터링 과정과 연관된 데이터 수집을 수행하기 위해 접촉되는 예외 식별자(262), 연관된 임계값(264), 데이터 수집을 위한 데이터 식별자 및 구성요소 타입(266) 및 시스템 구성요소(268)을 포함한다.

예외 식별자 262는 도 6의 시스템 210에서 예외 또는 트리거 이벤트를 고유하게 식별하는 식별자이다. 각 예외 조건과 연관된 것은 임계값 264이다. 도 11의 테이블에서, 임계값은 시간 값이거나 예외 식별자와 연관되어 정해진 다른 임계값일 수 있다. 예를 들어, 예외 식별자는 요청을 서비스하는데 요구되는 시간의 양과 관련되고, 5초와 같은 임계값은 알람 조건 또는 트리거 이벤트가 클라이언트 프로세스로 실행되는 대리자에 의해 검출되는 최대 기간을 특정할 수 있다. 열 266은 다양한 데이터 식별자를 계산하거나 이들을 보고하는 역할을 담당하는 상이한 구성요소 타입에서 수집될 데이터 아이템들을 식

별하는 다양한 데이터 식별자들을 특정한다. 예를 들어, 데이터 식별자 'A'는 이러한 특수한 예외와 관련된 각각의 서버들 또는 서버 프로세스들에 의해서 계산될 수 있다. 데이터 식별자 'B' 및 'C'는 이러한 클라이언트 서버 접속에 관여하는 각각의 클라이언트들로부터 보고되거나 수집될 수 있다.

데이터 수집을 위한 시스템 구성요소들은 열 268에 정해져 있다. 예를 들어, 열 268에서 정해진 각각의 클라이언트 시스템들은 열 266에 있는 대응하는 데이터 식별자 파라미터를 생성하여 제공하기 위해 접촉된다. 예를 들어, 만약 열 268에서 서버 4와 서버 5가 모두 확인되면, 이들 양자의 서버들은 열 266에 분류된 파라미터 'A'에 대한 값을 공급하기 위해 접촉된다. 마찬가지로, 열 268에서 만약 클라이언트 1이 정해지고, 열 266에서 파라미터 'B' 및 'C'가 확인되면 클라이언트 1은 파라미터 'B' 및 'C'에 대한 값을 위해 접촉된다. 컨트롤러는 테이블 260에 저장되어 있는 정보를 이용할 수 있는데, 예를 들어, 어떠한 파라미터 값에 대해서 어떠한 클라이언트 및 서버 시스템이 접촉되어야 하는지를 결정하는 경우에 이용할 수 있다. 컨트롤러는, 예를 들어, 테이블의 열 268을 조사하여 모니터될 특정 접속에 대한 데이터를 수집하기 위해 어떤 클라이언트 및 서버 시스템과 접촉해야 되는지를 결정한다. 열 266에 저장되어 있는 데이터는 컨트롤러가 열 268에 정해진 각각의 클라이언트-서버 시스템으로부터 어떤 파라미터 또는 변수를 받기를 기대하는지 확인한다.

도 12를 참조하면, 도 12는 도 6의 시스템에서 실행되는 분산 어플리케이션의 모니터링시 수집된 상이한 데이터 파라미터들에 대한 다양한 캐시 값(cached values)의 테이블의 일실시예의 예시도이다. 테이블 280의 캐시 값은 데이터 수집 식별자 282, 수집될 데이터의 설명 284, 및 캐시 값과 데이터 값과 관련된 시스템의 구성요소 286을 포함한다. 예를 들어, 데이터 수집 식별자 A는 클라이언트 CPU 시간을 설명한다. 표 280의 행 1에서, 3초의 캐시 값은 클라이언트 1의 파라미터 A와 연관된다. 유사한 통계량이 클라이언트 2의 CPU 시간과 관련해서 열 2에 적혀있다.

하나의 구현에에서, 이러한 값들의 캐싱(caching)은 컨트롤러에 의해 모니터 되는 컨트롤러 변수들 또는 통계량들에 대해서만 저장된다. 다른 구현예에서, 캐시되는 통계량들은 클라이언트 및 서버에 대해서 도 12의 테이블 280에 저장되어 있는 것과 같은 것들을 포함한다. 이들과 같은 통계량들은 컨트롤러 216에 의해 유지될 수 있다. 이들 값들은 데이터 저장소 220에 저장될 수 있고 도 6의 시스템의 트래픽을 최소화하기 위해 컨트롤러 216에 의해 엑세스되고 관리될 수 있다. 다른 구현예에서, 각각의 클라이언트들은 물론 서버들은 컨트롤러에 대해서뿐만 아니라 다양한 변수 또는 데이터 수집 파라미터들에 대한 데이터 저장소에 엑세스하여 정보를 갱신할 수 있다. 그러나, 이러한 구현에에서는, 네트웍에 대한 트래픽의 양을 최소화하여 도 6의 컴퓨터 시스템 210에서 실행되는 어플리케이션들의 모니터링시 보다 효과적인 데이터 수집을 가능하게 하기 위해서, 테이블 280에 저장된 캐싱 정보(caching Information)와 관련된 데이터 저장소 220에는 컨트롤러만이 엑세스해서 관리하는 것이 바람직하다. 도 12의 테이블 280에 포함된 것들과 같은 통계량들은 콘솔 218에 의한 것과 같이 정보를 보고하는데 이용될 수 있다는 것을 유의해야 한다.

도 11의 필드 264와 함께 특정된 임계값들은 시간에 기초한 것이거나 수집될 데이터의 타입에 따른 다른 데이터 양일 수 있다. 따라서, 데이터 저장소 220에 저장되어 있는 다양한 데이터 값 및 파라미터들은 데이터 상의 변화가 존재한다면 클라이언트들 및 서버들에게 전파된다. 전술한 바와 같이, 각각의 클라이언트들 및 서버들은 필요한 경우 데이터 및 파라미터들에 접근한다. 데이터 집합은 도 11 및 도 12의 테이블에 저장된 것과 같은 완전한 데이터 집합을 포함하거나 다양한 클라이언트 및 서버 파라미터들과 관련된 데이터 집합의 일부를 포함할 수 있다.

도 11의 테이블 260에 포함된 데이터는 도 6의 컴퓨터 시스템 210의 물리적인 레이아웃과 관련된 구조 정보(configuration information)를 포함한다. 이러한 구조정보와 같은 데이터는 일반적으로 컨트롤러 프로세스 216에 의해서 읽히기 위해서만 필요하다. 서버 또는 클라이언트에서 수집되는 데이터에 관한 파라미터들과 같은 다른 데이터는 구조 정보와 같은 데이터와는 개념적으로 구분되고 클라이언트 및 서버 프로세스들의 필요에 따라 다른 장소에 저장된다.

일반적으로, 도 6의 시스템 210은 대리자로부터 서버 데이터를 원격 수집하는 방법을 설명하고 있다. 도 6의 클라이언트 및 서버상의 대리자 소프트웨어에 의해 실행되는 것과 같은, 이러한 원격 데이터 수집 능력(remote data collection facility)은 다양한 클라이언트 및 서버의 다른 소프트웨어를 통해서 제공될 수 있다. 예를 들어, 특정 데이터베이스에서, 특정 데이터 수집 능력은 데이터베이스의 어플리케이션 프로그래밍 인터페이스(API)에 포함될 수 있다. 동일하게, 데이터베이스의 트랜스액션 정보에 대한 그러한 능력이 API를 통해서 데이터베이스에 의해서는 제공되지 않으면, 이러한 능력들은 컨트롤러에게 필요한 데이터를 수득하고 수집하기 위해서 클라이언트 및 서버상의 215a-215n 및 217a-217n과 같은 대리자 프로세스(agent processes)에 의해 제공될 수 있다.

도 7과 관련하여 앞에서 설명했던 클라이언트 및 서버상의 대리자 프로세스의 구현예는 트리거 이벤트 또는 예외 조건의 발생시 데이터 수집 및 성능 모니터링을 제공한다는 것을 유의해야 한다. 또한, 데이터 수집은 특정 트리거 이벤트 발생시 일어나지 않고 주기적으로 수행될 수 있다. 즉, 예를 들어, 도 7을 다시 참조하면, 제 232 단계에서 이벤트 트리거링 및 검출 대신에, 대기 루프를 실행하고, 소정의 시간 증분에 따라 정기적으로 정해진 트리거 이벤트 및 데이터를 수정할 수 있다. 주기적으로 데

이터를 수집하는 것은, 예를 들어, 특정한 시간 증분에 따라 도 6의 시스템 210의 다양한 측면의 '스냅샷(snapshot)'에 따라 사용 통계를 수집하는데 이용될 수 있다.

특수한 구현에에서, 도 6의 시스템은 212a 내지 212n과 같은 하나 이상의 클라이언트 프로세스들을 가질 수 있는데, 이러한 프로세서들은 마이크로소프트사의 윈도우 95, 98 또는 윈도우 NT중 하나인 운영체제를 갖는 컴퓨터 시스템에 운영체제를 갖는 컴퓨터 시스템에서 실행될 수 있다. 도 6과 관련된 구현예에서, 클라이언트와 서버 프로세스들 및 컨트롤러 프로세스가 실행되는 각각의 컴퓨터 시스템들은 TCP 프로토콜을 이용해서 통신할 수 있다.

방금 설명한 기술 및 구현예들은 컴퓨터 시스템에서 분산 어플리케이션 들의 성능을 모니터 하고 관리하는 방법을 제공한다. 이러한 기술은 어떤 조건이 일어났을 때 트리거링 이벤트 개시에 기초하거나, 주기적으로 도 6의 컴퓨터 시스템 210에 있는 하나 이상의 분산 어플리케이션 구성요소들로부터 데이터가 수집된다. 전술한 바와 같이, 구성요소들은 클라이언트 및 서버 컴퓨터 시스템들은 물론 각각의 컨트롤러 구성요소 및 데이터베이스 서버에 대한 전용 프로세서를 포함할 수 있다.

모니터 링 데이터는 원격 방식으로 수집되거나 시스템 성능을 분석하거나 문제를 진단하는데 이용될 수 있다. 전술한 시스템은 다양한 클라이언트 서버 구성요소들로부터 데이터를 수집하는데 요구되는 오버헤드를 최소화하기 위한 의도로 메인 코디네이터로서 컨트롤러를 포함한다. 컨트롤러는 컴퓨터 시스템 210에 있는 상이한 구성요소들로부터 데이터 수집 아이템들(data collection items)에 대한 요청을 조정한다. 도 6의 데이터 저장소 220에 수집되고 저장되어 있는 데이터통계량 및 정보는 분석되고 콘솔 218을 통해서 사용자에게 보고되어 원인을 확인하고 시스템 성능과 관련된 정보를 디스플레이하는 것을 도울 수 있다. 데이터 수집 능력상 이러한 실시간 이벤트는 도 6의 컴퓨터 시스템 210에서 실행되는 분산 어플리케이션들의 성능을 모니터링하는 유연한 기술을 제공한다. 이러한 원격 데이터 수집 능력은 분산 컴퓨터 시스템 환경뿐만 아니라 다른 분야에서도 적용될 수 있다는 것을 이해하여야 한다.

하나의 구현에에서, 상이한 스레드들(threads)이 도 7과 함께 설명된 상이한 태스크들을 수행하기 위하여 대리자 프로세스의 컨텍스트에서 실행될 수 있다. 예를 들어, 자원 수집 스레드(resource gathering thread)는 클라이언트의 모든 자원 변수 또는 파라미터들에 대한 값들을 계산할 수 있다. 주기적 평가 스레드(periodic evaluation thread)는 소정의 시간 간격에 따라 특정된 클라이언트 태스크를 평가할 수 있다. 비-주기적 평가 스레드는 모니터되는 클라이언트-서버 접속에 어떤 이벤트가 있을 때 어플리케이션 태스크를 평가할 수 있다. 앞의 설명에서 스레드의 이용은 운영체제 및 도 6의 컴퓨터 시스템에 포함되어 있는 각각의 상이한 프로세서들에서 실행되는 다른 소프트웨어에 의해 제공된 기능에 따른다.

하나의 구현에에서, 모니터 되는 클라이언트와 서버들은 물론 타임 버짓, 평가될 각 서버에 대한 로그인 및 패스워드 정보는 데이터 저장소 220에 저장되는 것과 같은 구조 정보(configuration information)에 포함된다. 사용자는, 예를 들어, 이러한 구조 정보를 콘솔 218을 통해서 특정할 수 있다.

태스크 또는 어플리케이션 은 그것이 클라이언트 및/또는 서버 그리고 그들 사이의 접속이 평가되어야 한다면 클라이언트 및/또는 서버에 '부착(attached)'되거나 결합될 수 있다. 일반적으로,어플리케이션의 경우, 명칭, 다양한 예외들, 조건들 및 하나 이상의 액션들 및 상태 변수들이 특정될 수 있다. 어플리케이션은 어떤 기계 및/또는 서버들 및 접속들이 평가되어야 하는지를 특정함으로써 특정한 접속에 부착되거나 결합된다. 클라이언트 어플리케이션은 그러한 클라이언트 기계 위에서 실행되는 대리자가 그러한 어플리케이션을 평가하는 클라이언트 기계에 부착될 수 있다. 어플리케이션은 클라이언트 및 서버에 부착될 수 있는데, 여기서 예를 들어, 대리자 또는 그러한 대리자의 비-주기적 평가 스레드가 특정 클라이언트-서버 접속과 관련된 활동이 있을 때 그러한 어플리케이션을 평가한다. 서버 어플리케이션은 다양한 서버 프로세서들에서 실행되는 대리자가 어플리케이션 을 주기적으로 평가하는 서버에 부착될 수 있다.

콘솔 218의 하나의 구현에는 자동 데이터 수집을 제공함은 물론 파라미터를 수정하고 주기적인 평가의 시간 간격을 특정하거나, 비즈니스 트랜스액션 파일의 생성 및 보존과 같은, 사용자에게 특수한 옵션을 허락하는 것과 같은 사용자 인터페이스에 대해 다른 편의를 제공한다는 것을 유의해야 한다.

몇몇 구현예에서는, 트리거링 데이터와 '트랜스액션'으로 간주되는 어플리케이션 에 의해 수행되는 다양한 어플리케이션 들 또는 오퍼레이션들을 갖는 시스템에 의해 수집되는 데이터를 결합시킬 수 있다.

도 13을 참조하면, 이러한 어플리케이션 비즈니스 트랜스액션의 모니터 링을 용이하게 하는 비즈니스 트랜스액션 시스템 300은 특수한 네트웍 정보와 특정 어플리케이션 비즈니스 트랜스액션을 연관시키는 정의 및 규칙을 가리키는 데이터를 포함한다. 비즈니스 트랜스액션 명세 302를 포함한다.

비즈니스 트랜스액션 명세 데이터 302는 비즈니스 트랜스액션 명세 302와 모니터 된 네트웍 및 다른 활동을 이용하여 사용자 데이터, 트랜스액션 데이터 및 예외들을 만드는 트랜스액션 엔진 304에 제공된다. 후술하는 바와 같이, 트랜스액션 엔진 304는 비즈니스 트랜스액션 명세 302에 기술된 비즈니스 트랜스액션과 관련되는 네트웍으로부터의 모니터된 활동의 부분들을 인식할 수 있다. 비즈니스 트랜스액션 명세 302에 그와 함께 제공된 관련(association) 및 규칙들에 기초하여, 트랜스액션 엔진 304는 데이터 저장소 220에 저장되는 어플리케이션 트랜스액션 특이적 사용(application transaction specific usage), 트랜스액션 데이터, 및 예외들을 만든다.

모니터 되는 활동(activity)은 일반적으로 네트웍을 포함하고, 사용자 인터페이스 활동, 원격 시스템 활동 및 시스템 프로세스 활동과 같은 다른 활동을 포함한다는 것을 유의해야 한다. 사용자 인터페이스 활동은 예를 들어 마우스 장치에 의한 버튼의 선택과 같은, 사용자 인터페이스와 관련된 사용자 개시 활동(user initiated activity)일 수 있다. 원격 시스템 활동은, 예를 들어, 도 6의 시스템 210과 원격적으로 접속된 다른 컴퓨터 시스템과 관련된 활동을 포함할 수 있다. 시스템 프로세스 정보는, 예를 들어, CPU 사용 정보를 포함할 수 있다.

도 14를 참조하면, 흐름도 320은 도 13의 트랜스액션 엔진 304의 기본적인 동작을 설명한다. 제 322 단계에서, 트랜스액션 엔진은 네트웍으로부터 모니터된 활동에 대응하는 데이터를 수집한다. 제 322 단계에 이어서 테스트 단계 제 324 단계에서는 수집된 데이터들이 비즈니스 트랜스액션 명세 302의 규칙에 적합한지를 판단한다. 비즈니스 트랜스액션 명세 302의 규칙들은 매 어플리케이션 단위로 정해질 수 있고 수집된 데이터들이 각 규칙에 순차적으로 매치되도록 특정한 순서대로 열거될 수 있다. 일단 어플리케이션이 결정되면, 수집된 데이터는 네트웍 데이터와 연관되는 어플리케이션과 매치되는 규칙의 부분을 찾음으로써 비즈니스 트랜스액션 명세 302내의 규칙들에 매치될 수 있다.

데이터는 네트웍 패킷 정보에 포함되는 목적지의 포트 번호와 IP 주소를 조사함으로써 특정 어플리케이션과 연관될 수 있다는 것을 유의해야 한다. 각 시간에 새로운 포트가 생성되고, 어플리케이션, 포트 및 IP 주소를 가리키는 네트웍 정보는 테이블에 포함된다. 테이블을 이용하여, 테이블 내에서 포트 번호와 IP 주소를 찾아봄으로써 네트웍 데이터를 특정한 어플리케이션과 매치시킬 수 있다. 프로세스 ID는 동일한 포트를 통해서 동일한 어플리케이션에 엑세스하는 상이한 이용자들을 서로 구별하는데 이용될 수 있다는 것을 유의해야 한다. 따라서, 사용자 1 및 사용자 2가 모두 포트 B를 이용해서 어플리케이션 A에 엑세스하면, 네트웍 데이터에 대한 프로세스 ID들을 서로 구별함으로써 사용자 1에 대한 트랜스액션은 사용자 2에 대한 트랜스액션과 구분되어 추적될 것이다. 본원의 다른 부분에서도 언급되는 바와 같이, 프로세스 ID들, 포트 ID들 및 데이터의 IP 주소들은 대리자에 의해 수득되고 제공된다는 것을 유념해야 한다.

제 322 단계에서 일어나는 데이터 수집은 트리거 이벤트의 발생과 함께 수행된다는 것을 유의해야 한다. 하나의 경우에, 트리거 이벤트와 관련된 데이터는 트랜스액션 완료되고 소정 양의 시간과 같은 임계값을 초과하는 경우에 수집될 수 있다. 다른 경우에, 트리거 이벤트와 관련된 데이터는 트랜스액션이 완료되지 않았을 때 수집될 수 있다. 이러한 다른 경우에, 데이터 수집이 일어나고 트랜스액션은 첫 번째 경우와 유사한 방식으로 처리된다.

제 324 단계에서 데이터가 특정 규칙에 부합되지 않는 것으로 판단되면, 제 322 단계로 리턴하여 더 많은 데이터를 수집한다. 그렇지 않고, 제 324 단계에서 데이터가 특정 규칙에 부합되는 것으로 판단되면, 제 324 단계로부터 제 326 단계로 진행하여 트랜스액션 엔진 304는 비즈니스 트랜스액션 정보를 생성하고 저장한다.

제 326 단계에 이어, 제 328 단계에서는 비즈니스 트랜스액션 명세 302에서 임계값을 초과하는지를 판단한다. 데이터를 특정 어플리케이션과 연관시키는 것 이외에, 비즈니스 트랜스액션 명세 302는 다양한 트랜스액션에 대한 특정 임계값을 나타내는 정보를 포함한다. 예를 들어, 비즈니스 트랜스액션 명세 302는 첫 번째 동작(operation)에 이어 두 번째 동작이 소정의 양의 시간 내에 수행되어야 한다는 것을 가리킬 수 있다. 만약 첫 번째 동작이 일어났지만 두 번째 동작이 지연되는 경우, 임계값을 초과할 것이다. 이러한 타입의 동작의 예들은 데이터의 요청, 데이터의 수신, 또는 두 번째 윈도우의 오픈 요청이 수반되는 첫 번째 윈도우에 대한 오픈 요청을 포함한다.

제 328 단계에서, 임계값을 초과하지 않는 것으로 판단되면, 제 328 단계로부터의 제어 경로는 다시 제 322 단계로 진행하여 추가의 네트웍 데이터를 수집한다. 그렇지 않고, 제 328 단계에서 임계값을 초과하는 것으로 판단되면, 제 328

단계로부터의 제어 경로는 제 330 단계로 진행하여 임계값을 초과했다는 사실을 콘솔 및 컨트롤러에 보고하여 본원에 기술된 바와 같은 적당한 액션을 취한다.

도 15를 참조하면, 흐름도 350은 데이터가 비즈니스 트랜스액션 명세 302의 규칙에 부합되는지를

판단하는, 도 14의 제 324 단계의 더욱 상세한 단계들을 도시한 것이다. 제 352 단계에서, 규칙을 읽어들여 처리한다. 규칙들은 게게의 규칙과 동작들 및 그의 시험들을 나타내는 특정 신텍스를 갖는 텍스트 파일을 제공하는 것을 포함하는 다양한 종래의 방식중 어느 하나의 방식에 의해 특정될 수 있다.

제 352 단계에 이어, 제 354 단계에서 규칙이 처리될 데이터와 매치되는지 판단한다. 만약 그렇다면, 제 354 단계로부터의 제어 경로는 제 356 단계로 진행하여 루틴으로부터 빠져나간다. 그렇지 않고, 제 354 단계에서 데이터가 처리될 특정한 규칙(즉, 규칙이 제 352 단계에서 페치)에 부합되지 않는 것으로 판단되면, 제 354 단계로부터의 제어경로는 제 358 단계로 진행하여 생략시 규칙(default rule)이 적용되어야 하는지 판단한다. 제 358 단계에서, 생략시 규칙이 적용되어야 하는 것으로 판단되면, 제 358 단계로부터의 제어 경로는 제 360 단계로 진행하여 루틴으로부터 빠져나간다. 그렇지 않고, 제 358 단계에서 생략시 규칙이 적용되지 않는 것으로 판단되면, 제 358 단계로부터 제 352 단계로 리턴하여 다음 규칙을 처리한다. 하나의 구현예에서, 생략시 규칙은 비즈니스 트랜스액션 명세 302에서 마지막 규칙이 될 수 있다.

도 16을 참조하면, 비즈니스 트랜스액션 시스템 300과 대리자 215a 사이의 관계를 도시한 개략도이다. 대리자 215a가 데이터를 모니터링할 때, 그러한 데이터는 상술한 바와 같이 비즈니스 트랜스액션 명세 302를 이용하여 사용, 트랜스액션 데이터 및 예외를 데이터 저장소 220에 제공하고 전술한 바와 같이 예외를 콘솔과 컨트롤러에 제공하는 비즈니스 트랜스액션 엔진 304으로 전달된다.

도 17을 참조하면, 워크스테이션 364는 비즈니스 트랜스액션 명세 302를 편집하고 생성하는데 이용되는 사용자 인터페이스 366을 포함한다. 비즈니스 트랜스액션 명세 302는 종래의 텍스트 파일로 구현될 수 있다. 이 경우에, 워크스테이션 364와 인터페이스 366은 종래의 컴퓨터 및 워드프로세서에 의해 제공된다. 대안으로, 비즈니스 트랜스액션 명세 302는 다른 종래의 수단에 의해 특정될 수 있다.

비즈니스 트랜스액션 명세 302가 텍스트 파일로 구현되는 실시예에서는, 몇몇 경우에 트랜스액션 304가 초기화시에 비즈니스 트랜스액션 명세 302를 읽고나서 더욱 효과적으로 엑세스될 수 있고 본원에서 설명한 기능들을 제공할 수 있는 실행 시간 동적 자료 구조를 생성하는 것이 가능하다.

하나의 구현예에서 비즈니스 트랜스액션 명세 302는 도 6의 데이터 저장소 220에 저장될 수 있다.

비즈니스 트랜스액션 명세 302를 제공하는 하나의 가능한 신텍스는 다음과 같다:

```
<BTSpecification>      -> <BTLangVersion> <BTApplicationSpec>+
<BTLangVersion>        -> Version <number>.<number>
<BTApplicationSpec>    -> Application <string-literal> <HeaderInfo> <BTRule>+
                          <ProcessRule>*
<HeaderInfo>           -> [ <HeaderEntry>+ ]
<BTRule>               -> BT <string-literal> <RuleClauses> END_BT
<ProcessRule>          -> PROCESS <ProcClause>+ END_PROCESS
<ProcClause>           -> BT <string-expr>
<HeaderEntry>          -> (<NumericEntry> <number>)* (<StringEntry> <string-literal>)*
<NumericEntry>         ->   NoiseThreshold              |
                            MaxTransactionTime          |
                            GuiProcess                  |
                            DiscardTimedOutTransaction  |
<StringEntry>          ->   WindowsToIgnore             |
                            ClassesToIgnore
<RuleClause>           -> <WindowClause>* <CommandClause>* <GeneralClause>*
```

```
<WindowClause>      ->  WindowStart          |
                        WindowEnd            |
                        WindowEndNew         |
                        WindowPrevious       |
                        WindowContains       |
                        StatusText           |
                        MenuCommand          |
                        WindowFilter         |
                        ButtonClick
                        <string-expr>
<WindowClause>      ->  WindowIgnore         |
                        WindowList
                        <string-list>
<CommandClause>     ->  CommandStart         |
                        CommandEnd           |
                        CommandContains
                        <command> <params>
<CommandClause>     ->  CommandList
                        <command-list>
<command>           ->  Select               |
                        Update               |
                        Delete               |
                        Execute              |
                        URL                  |
                        Insert               |
                        BeginTransaction     |
                        EndTransaction
<GeneralClause>     ->  Max  (BytesSent | BytesReceived |        roundTrips |

                    Operations )   (<number>) |
                     Min  (BytesSent | BytesReceived |       roundTrips |

                    Operations )   (<number>)     |
                      SetName   <string-literal>

<string-expr>       ->  <string-expr> OR  <string-expr>
<string-expr>       ->  NOT (<string-expr>)
<string-expr>       ->  <string-literal>
<string-list>       ->  <string-list> , <string-literal> | <string-literal>
<string-literal>    ->  'string'    |
                        'string' ( <params> )
<params>            ->  <params> , <param> | 'string'
```

코멘트는 첫 글자가 '#'인 모든 줄이다.

본 발명이 효과적으로 상술한 목적을 달성할 수 있음을 확인할 수 있다. 더욱이, 본 발명의 정신을 벗어나지 않는 범위 내에서 다양한 치환, 부가, 및 변형들이 이루어질 수 있고 본원에서 설명 및 묘사된 구현예들은 단지 설명의 목적을 위한 것으로 제한적인 의미를 갖는 것으로 해석되어서는 안된다: 후술하는 청구범위에 의해 한정되는 본 발명의 범위는 명세서의 용이상 최광의로 해석되어야 한다.

(57) 청구의 범위

청구항 1

트리거 이벤트들과 수집될 관련 데이터를 정의하는 단계:

클라이언트와 제 1 서버 사이의 접속을 모니터하면서 클라이언트에서 트리거 이벤트들중 하나의 발생을

검출하는 단계;

클라이언트에서 하나의 트리거 이벤트에 따라서 클라이언트 데이터를 수집하는 단계;

상기 하나의 트리거 이벤트의 발생의 검출을 컨트롤러에 통보하는 단계;

제 1 서버에게 상기 트리거 이벤트의 발생을 통보하는 단계;

상기 제 1 서버에 의해 제 1 서버 데이터를 수집하는 단계; 및

상기 제 1 서버 데이터를 컨트롤러로 전송하는 단계를 포함하는 분산 컴퓨터 시스템의 모니터링 방법.

청구항 2

제 1항에 있어서, 상기 방법이

상기 컨트롤러에 클라이언트 컨텍스트 정보를 전송하는 단계; 및

상기 클라이언트 컨텍스트 정보를 이용하여 상기 클라이언트로부터 수득되어야 하는 데이터를 결정하는 단계를 추가로 포함하는 방법.

청구항 3

제 2항에 있어서, 상기 방법이 상기 컨트롤러가 데이터 저장소에 포함되어 있는 데이터를 이용하여 어떤 데이터가 상기 클라이언트로부터 수득되어야 하는지 결정하는 단계를 추가로 포함하는 방법.

청구항 4

제 3항에 있어서, 상기 클라이언트 컨텍스트 정보가 상기 제 1 서버를 고유하게 식별하는 서버 식별자를 포함하는 방법.

청구항 5

제 1항에 있어서, 상기 방법이 상기 클라이언트에 의해 계산될 수 있는 상기 관련 데이터의 클라이언트 부분을 상기 클라이언트에 의해 수집하는 단계를 추가로 포함하는 방법.

청구항 6

제 5항에 있어서, 상기 클라이언트 부분이 상기 클라이언트와 상기 서버 사이의 접속에 관련된 정보를 포함하는 방법.

청구항 7

제 6항에 있어서, 상기 클라이언트 부분이 자원 메트릭(resource metric)을 포함하는 방법.

청구항 8

제 7항에 있어서, 상기 자원 메트릭이 프로세서 및 메모리 사용 정보를 포함하는 방법.

청구항 9

제 6항에 있어서, 상기 클라이언트 부분이 상기 접속에 관한 네트웍 대기시간 정보(network latency information)를 포함하는 방법.

청구항 10

제 1항에 있어서, 상기 클라이언트에서의 상기 데이터 수집이

상기 클라이언트에서 상기 데이터 수집을 수행하기 위한 기계 실행 프로그램을 특정하는 단계; 및

상기 기계 실행 프로그램을 실행하여 상기 클라이언트에서 상기 데이터를 수집하는 단계를 포함하는 방법.

청구항 11

제 1항에 있어서, 상기 방법이

하나 이상의 서버들에게 상기 트리거 이벤트의 발생을 통보하는 단계;

상기 하나 이상의 서버들에 의해 다른 서버 데이터를 수집하는 단계; 및

상기 다른 서버 데이터를 상기 컨트롤러로 전송하는 단계를 추가로 포함하는 방법.

청구항 12

제 1항에 있어서, 상기 방법이

상기 클라이언트 데이터를 상기 컨트롤러에 보내는 단계를 추가로 포함하는 방법.

청구항 13

제 1항에 있어서, 상기 제 1 서버 데이터가 상기 제 1 서버에 의해 처리될 현재 열려있는 트랜스

액션에 관한 정보를 포함하는 방법.

청구항 14

제 1항에 있어서, 상기 제 1 서버 데이터가 상기 제 1 서버에 의해 서비스될 요청에 관한 정보를 포함하는 방법.

청구항 15

제 1항에 있어서, 상기 제 1 서버 데이터가 상기 제 1 서버에 관한 사용 정보(usage information)를 포함하는 방법.

청구항 16

제 1항에 있어서, 상기 방법이 예외를 유발하는 트리거 이벤트에 관한 정보를 상기 컨트롤러에 의해 수집하는 단계를 포함하는 방법.

청구항 17

제 1항에 있어서, 상기 클라이언트 데이터의 수집이 상기 컨트롤러에 의해 원격적으로 이루어지는 방법.

청구항 18

제 1항에 있어서, 상기 분산 컴퓨터 시스템이 하나 이상의 클라이언트들 및 하나 이상의 서버들을 포함하고, 각각의 상기 하나 이상의 클라이언트들 및 하나 이상의 서버들이 상이한 컴퓨터 프로세서와 관련되고 상기 컴퓨터 프로세서에서 실행되는 프로세스인 방법.

청구항 19

제 18항에 있어서, 상기 컨트롤러가 전용 컴퓨터 프로세서에서 실행되는 프로세스인 방법.

청구항 20

제 1항에 있어서, 상기 분산 컴퓨터 시스템이 하나 이상의 클라이언트들과 하나 이상의 서버들을 포함하고, 컴퓨터 프로세서가 적어도 두 개의 클라이언트와 관련되고, 각각의 상기 두 개의 클라이언트들이 상기 컴퓨터 프로세서에서 실행되는 프로세스인 방법.

청구항 21

제 1항에 있어서, 상기 클라이언트 및 상기 제 1 서버가 각각 데이터를 수집하는 대리자 프로세스와 관련되어 있는 방법.

청구항 22

제 1항에 있어서, 상기 컨트롤러가 클라이언트 및 제 1 서버 데이터를 수집하는 코디네이터인 방법.

청구항 23

트리거 이벤트 및 수집될 관련 데이터를 정의하는 기계 실행 코드(machine executable code);

클라이언트와 제 1 서버 사이의 접속을 모니터 하면서 클라이언트에서 트리거 이벤트들중 하나의 발생을 검출하는 기계 실행 코드;

클라이언트에서 하나의 트리거 이벤트에 따라 클라이언트 데이터를 수집하는 기계 실행 코드;

컨트롤러에게 하나의 트리거 이벤트의 검출을 통보하는 기계 실행 코드;

상기 제 1 서버에게 트리거 이벤트의 발생을 통보하는 기계 실행 코드;

상기 제 1 서버에 의해 제 1 서버 데이터를 수집하는 기계 실행 코드; 및

제 1 서버 데이터를 컨트롤러에 전송하는 기계 실행 코드를 포함하는 분산 컴퓨터 시스템의 모니터링 시스템.

청구항 24

제 23항에 있어서, 상기 시스템이

상기 컨트롤러에게 클라이언트 컨텍스트 정보를 전송하는 기계 실행 코드; 및

상기 클라이언트 컨텍스트 정보를 이용하여 상기 클라이언트로부터 수득되어야 하는 데이터를 결정하는 기계 실행 코드를 추가로 포함하는 시스템.

청구항 25

제 24항에 있어서, 상기 컨트롤러가 데이터 저장소에 포함된 데이터에 엑세스하여 상기 클라이언트로부터 어떤 데이터를 수득할지를 결정하는 기계 실행 코드를 포함하는 시스템.

청구항 26

제 25항에 있어서, 상기 클라이언트 컨텍스트 정보가 상기 제 1 서버를 고유하게 식별하는 서버

식별자를 포함하는 시스템.

청구항 27

제 23항에 있어서, 상기 시스템이 상기 클라이언트에 의해 계산될 수 있는 상기 관련 데이터의 클라이언트 부분을 상기 클라이언트에 의해 수집하는 기계 실행 코드를 추가로 포함하는 시스템.

청구항 28

제 27항에 있어서, 상기 클라이언트 부분이 상기 클라이언트 및 상기 제 1 서버 사이의 상기 접속에 관련된 정보를 포함하는 시스템.

청구항 29

제 28항에 있어서, 상기 클라이언트 부분이 자원 메트릭을 포함하는 시스템.

청구항 30

제 29항에 있어서, 상기 자원 메트릭이 프로세서 및 메모리 사용 정보를 포함하는 방법.

청구항 31

제 28항에 있어서, 상기 클라이언트 부분이 상기 접속에 관한 네트웍 대기시간 정보를 포함하는 방법.

청구항 32

제 23항에 있어서, 상기 클라이언트에서 데이터를 수집하는 상기 기계 실행 코드가 상기 클라이언트에서 상기 데이터 수집을 수행하는 기계 실행 프로그램을 특정하는 기계 실행 코드를 포함하는 시스템.

청구항 33

제 23항에 있어서, 상기 시스템이

하나 이상의 다른 서버들에게 상기 트리거 이벤트의 발생을 통보하는 기계 실행 코드;

상기 하나 이상의 다른 서버들에 의해 다른 서버 데이터를 수집하는 기계 실행 코드; 및

상기 다른 서버 데이터를 상기 컨트롤러에 보내는 기계 실행 코드를 포함하는 시스템.

청구항 34

제 23항에 있어서, 상기 시스템이 상기 클라이언트 데이터를 상기 컨트롤러에 보내는 기계 실행 코드를 추가로 포함하는 시스템.

청구항 35

제 23항에 있어서, 상기 제 1 서버 데이터가 상기 제 1 서버에 의해 처리될 현재 열려있는 트랜스액션에 관한 정보를 포함하는 시스템.

청구항 36

제 23항에 있어서, 상기 제 1 서버 데이터가 상기 제 1 서버에 의해 서비스되는 요청에 관한 정보를 포함하는 시스템.

청구항 37

제 23항에 있어서, 상기 제 1 서버 데이터가 상기 제 1 서버에 관한 사용 정보를 포함하는 시스템.

청구항 38

제 23항에 있어서, 상기 시스템이 상기 클라이언트에 의해 예외를 유발한 트리거 이벤트에 관한 정보를 수집하는 기계 실행 코드를 추가로 포함하는 시스템.

청구항 39

제 23항에 있어서, 클라이언트 데이터를 수집하는 상기 기계 실행 코드가 상기 컨트롤러에 의해 원격 데이터 수집(remote data gathering)을 수행하는 시스템.

청구항 40

제 23항에 있어서, 상기 분산 컴퓨터 시스템이 하나 이상의 클라이언트들 및 하나 이상의 서버들을 포함하고, 각각의 상기 하나 이상의 클라이언트들 및 하나 이상의 서버들이 상이한 컴퓨터 프로세서와 관련되고 상기 컴퓨터 프로세서에서 실행되는 프로세스인 시스템.

청구항 41

제 40항에 있어서, 상기 컨트롤러가 전용 컴퓨터 프로세서에서 실행되는 프로세스인 시스템.

청구항 42

제 23항에 있어서, 상기 분산 컴퓨터 시스템이 하나 이상의 클라이언트들과 하나 이상의 서버들을 포함하고, 컴퓨터 프로세서가 적어도 두 개의 클라이언트와 관련되고, 각각의 상기 두 개의 클라이언트들이 상기 컴퓨터 프로세서에서 실행되는 프로세스인 시스템.

청구항 43

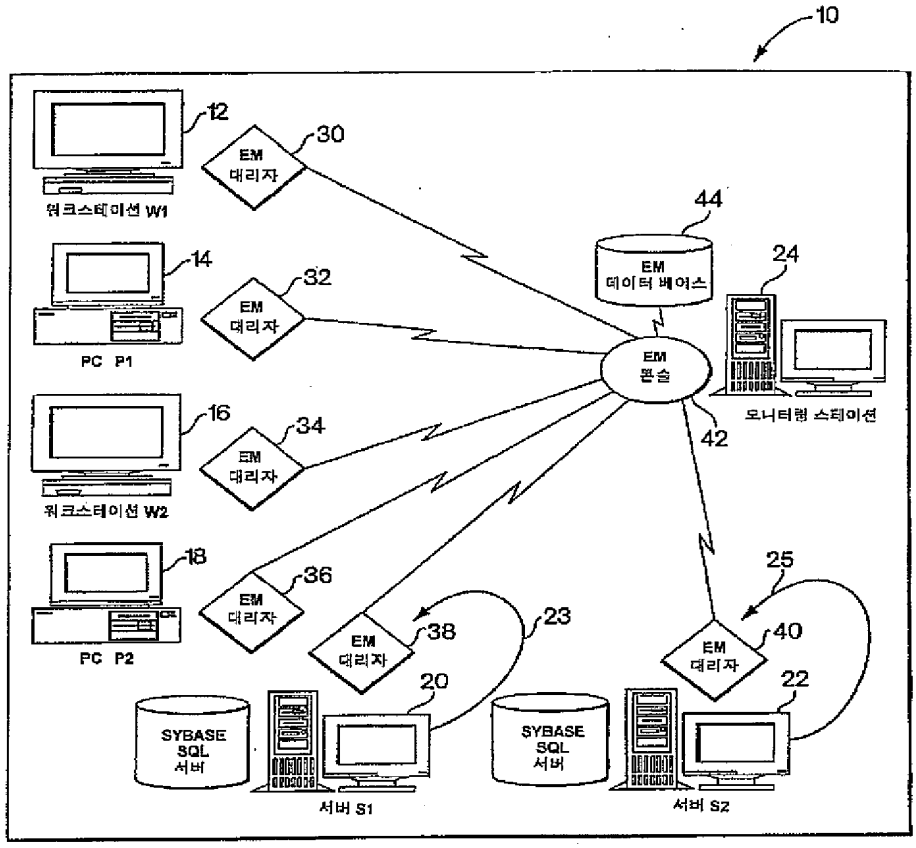
제 23항에 있어서, 상기 클라이언트 및 상기 제 1 서버가 각각 데이터를 수집하는 대리자 프로세스와 관련되어 있는 시스템.

청구항 44

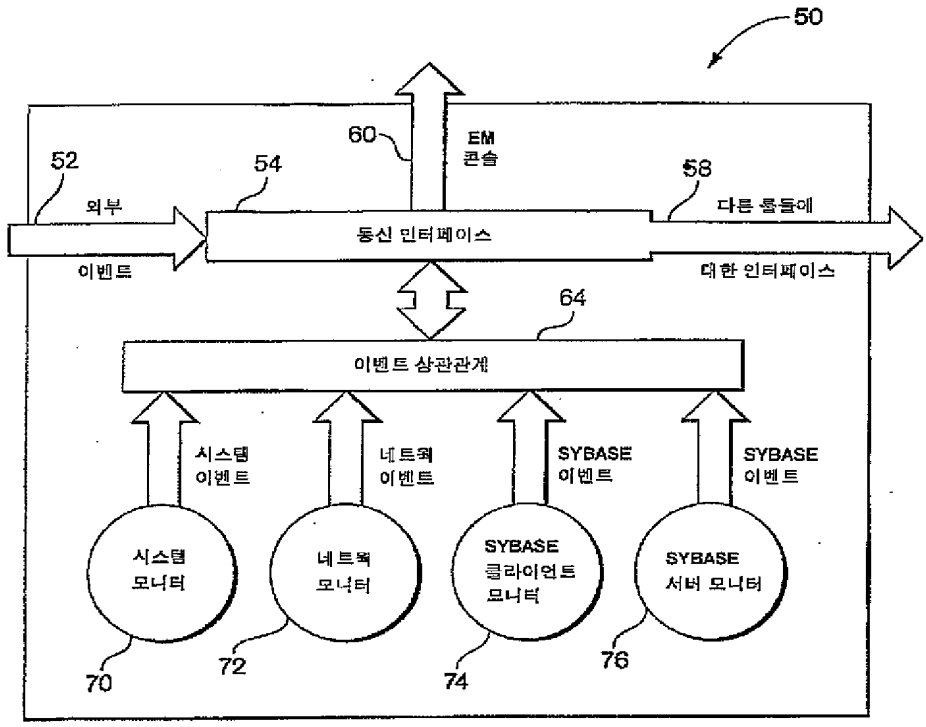
제 23항에 있어서, 상기 컨트롤러가 클라이언트 및 제 1 서버 데이터를 수집하는 코디네이터인 시스템.

도면

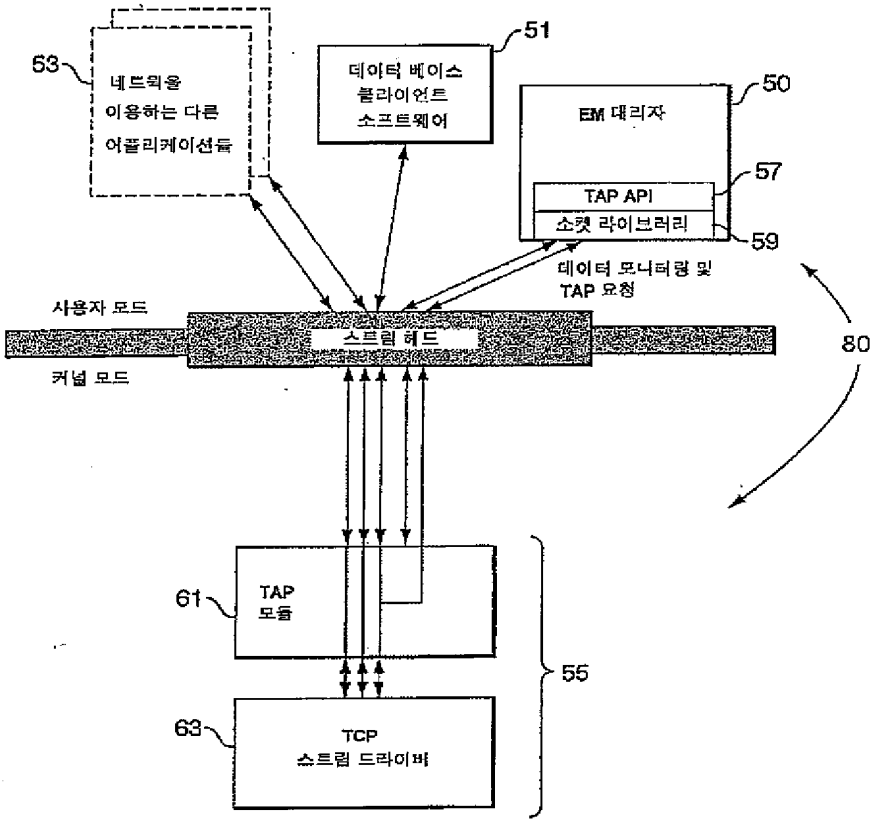
도면1



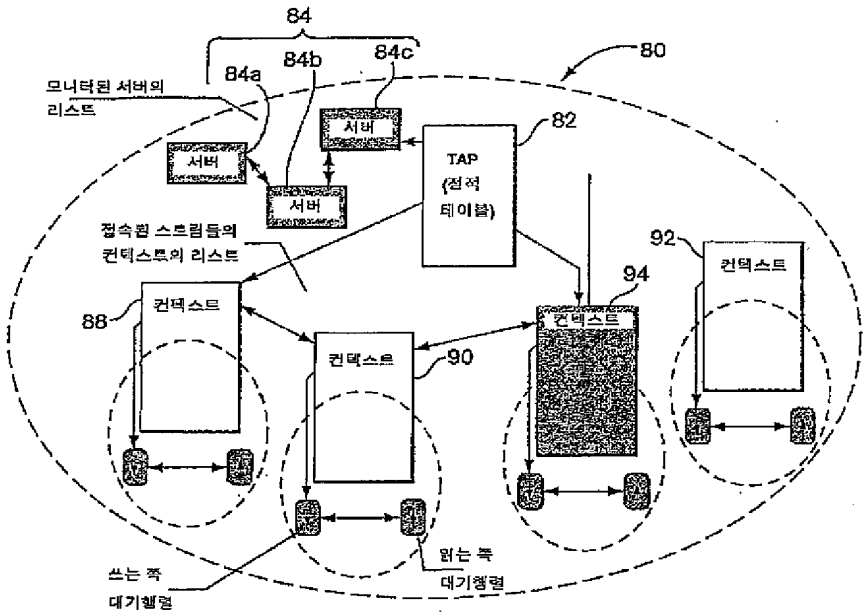
도면2


50
60
EM
콘솔
52
54
58
외부
다른 툴들에
이벤트
동신 인터페이스
대한 인터페이스
64
이벤트 상관관계
시스템
이벤트
네트웍
이벤트
SYBASE
이벤트
SYBASE
이벤트
시스템
모니터
네트웍
모니터
SYBASE
클라이언트
모니터
SYBASE
서버 모니터
70
72
74
76

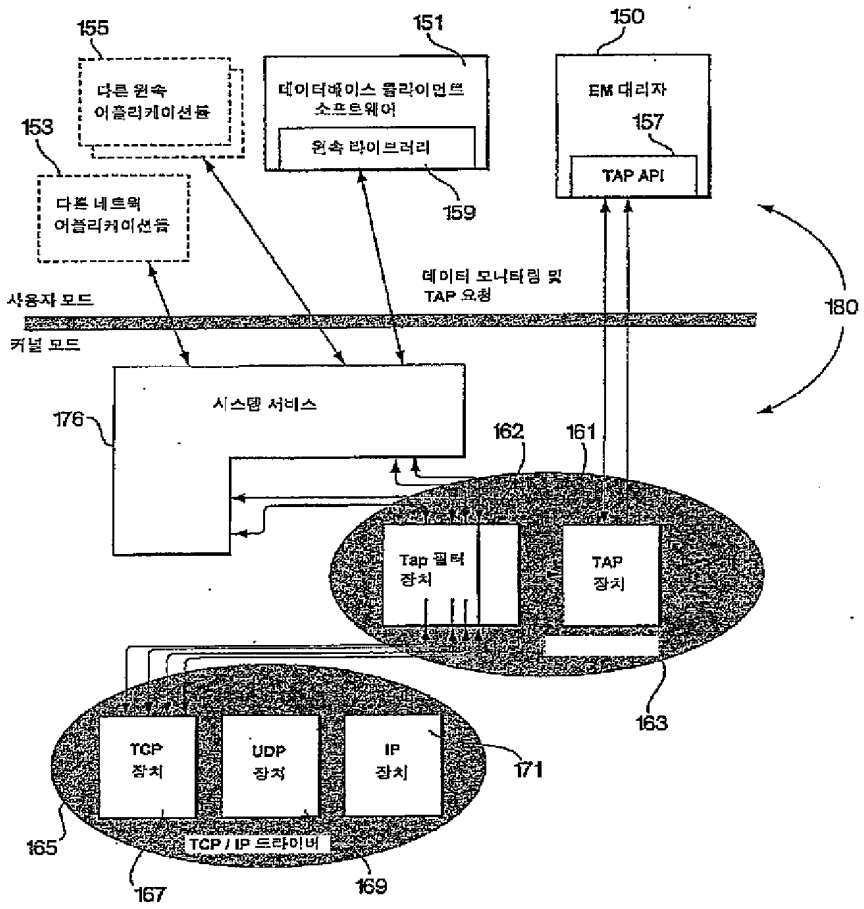
도면3


53
네트워을
이용하는 다른
어플리케이션들
51
데이터 베이스
클라이언트
소프트웨어
50
EM 대리자
57
TAP API
소켓 라이브러리
59
데이터 모니터링 및
TAP 요청
사용자 모드
스트림 헤드
커널 모드
80
61
TAP
모듈
55
63
TCP
스트림 드라이버

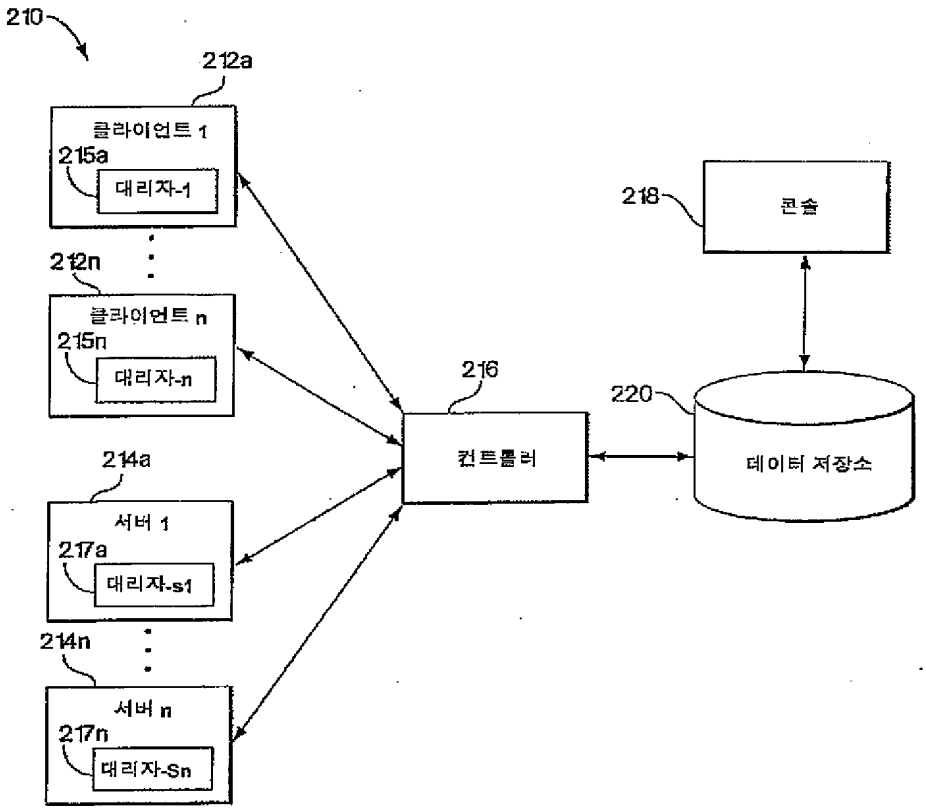
도면4


84
84a
84b
84c
80
모니터된 서버의
리스트
서버
서버
서버
TAP
(점적
테이블)
82
접속된 스트림들의
컨텍스트의 리스트
92
컨텍스트
94
88
컨텍스트
컨텍스트
컨텍스트
90
읽는 쪽
대기행렬
쓰는 쪽
대기행렬

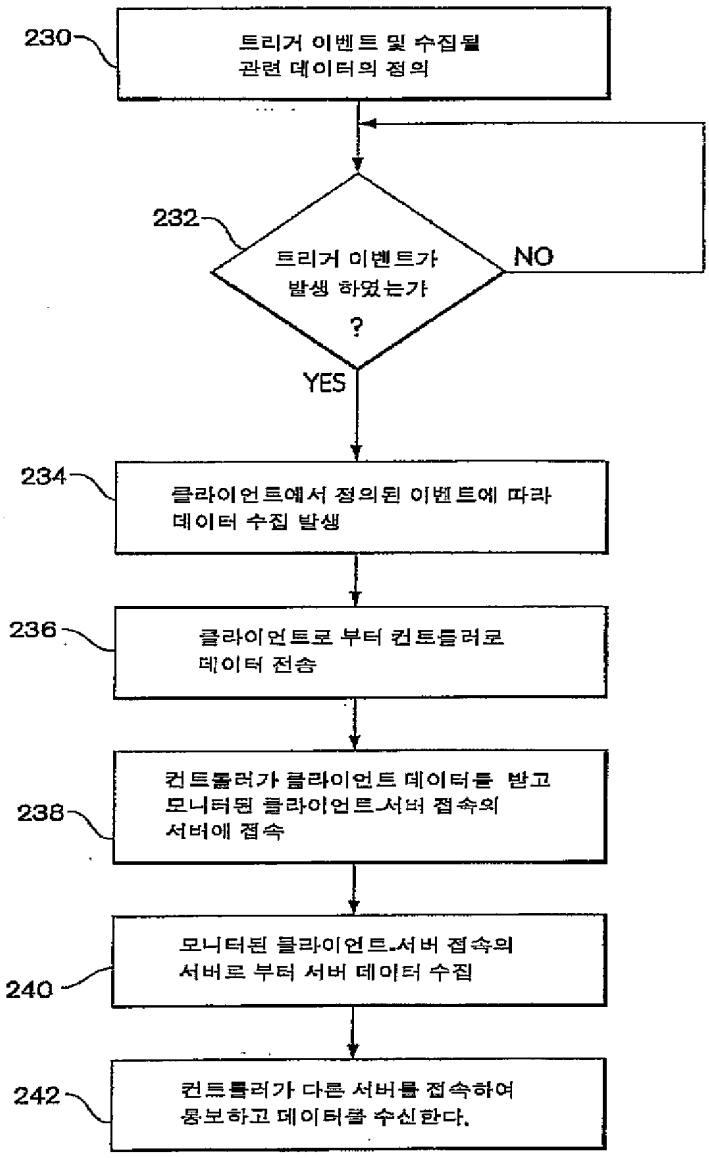
도면5


155
151
150
다른 원속
이플리케이션들
데이터베이스 클라이언트
소프트웨어
원속 라이브러리
EM 대리자
157
TAP API
153
다른 네트웍
어플리케이션들
159
데이터 모니터링 및
TAP 요청
사용자 모드
커널 모드
180
시스템 서비스
176
162
161
Tap 필터
장치
TAP
장치
163
TCP
장치
UDP
장치
IP
장치
171
165
TCP / IP 드라이버
167
169

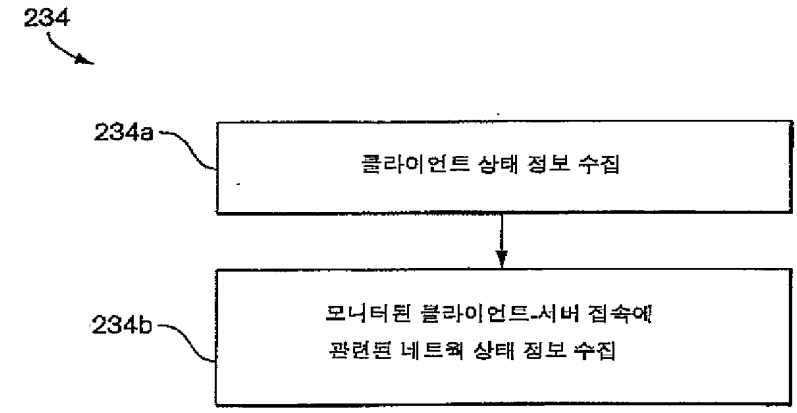
도면6


210
212a
클라이언트 1
215a
대리자-1
212n
클라이언트 n
215n
대리자-n
214a
서버 1
217a
대리자-s1
214n
서버 n
217n
대리자-Sn
216
컨트롤러
218
콘솔
220
데이터 저장소

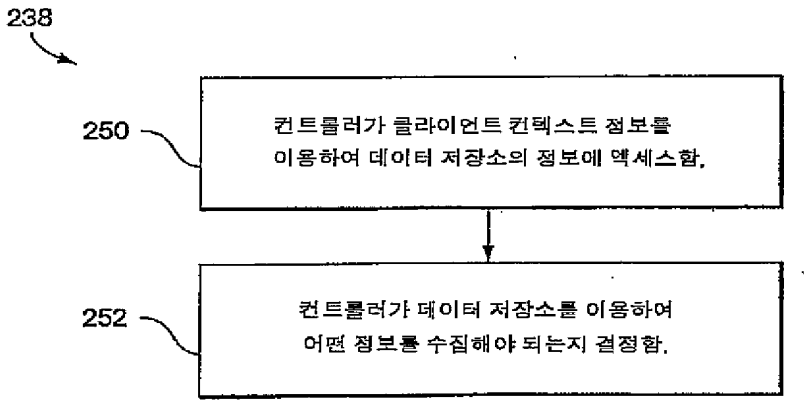
도면7


230
트리거 이벤트 및 수집될 관련 데이터의 정의
232
트리거 이벤트가 발생 하였는가 ?
NO
YES
234
클라이언트에서 정의된 이벤트에 따라 데이터 수집 발생
236
클라이언트로 부터 컨트롤러로 데이터 전송
238
컨트롤러가 클라이언트 데이터를 받고 모니터된 클라이언트-서버 접속의 서버에 접속
240
모니터된 클라이언트-서버 접속의 서버로 부터 서버 데이터 수집
242
컨트롤러가 다른 서버를 접속하여 통보하고 데이터를 수신한다.

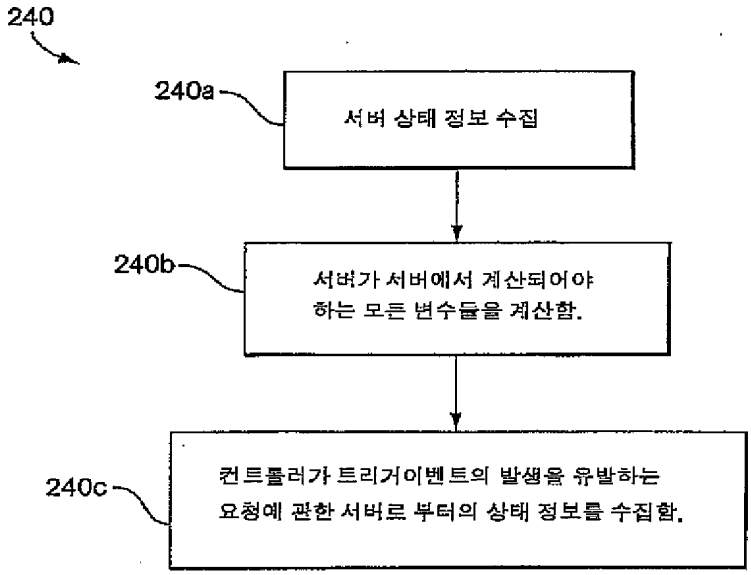
도면8


234
234a
클라이언트 상태 정보 수집
234b
모니터된 클라이언트-서버 접속에
관련된 네트웍 상태 정보 수집

도면9


238
250
컨트롤러가 클라이언트 컨텍스트 정보를
이용하여 데이터 저장소의 정보에 액세스함.
252
컨트롤러가 데이터 저장소를 이용하여
어떤 정보를 수집해야 되는지 결정함.

도면10



도면11

260

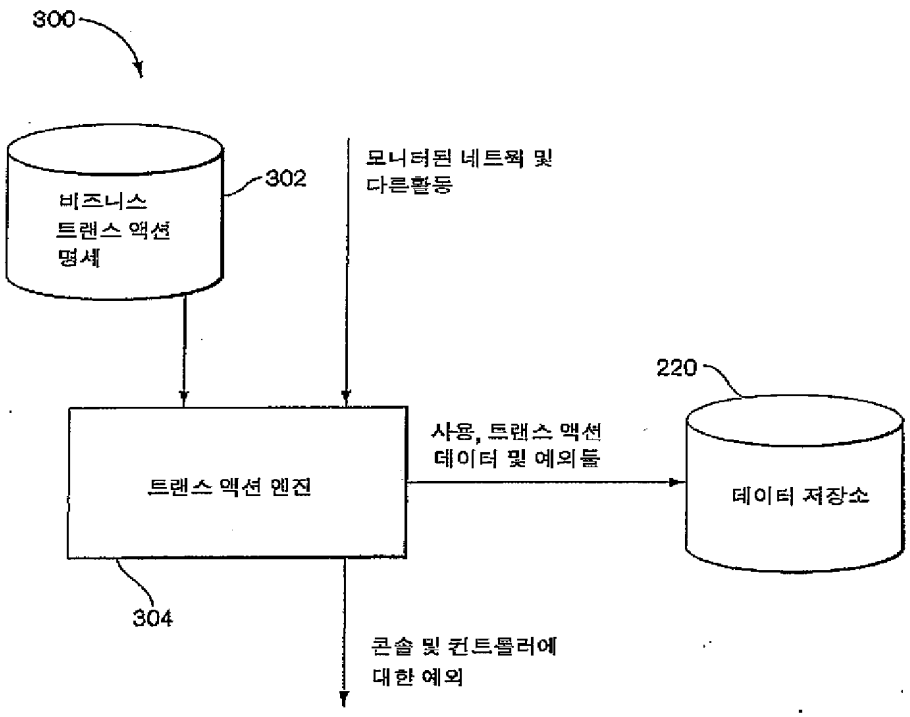
262 예외 ID	264 임계값	266 수집할 데이터 ID 및 구성요소 타입	268 데이터 수집을 위한 시스템 구성요소
A123	5 초	A, 서버 B, 클라이언트 C, 클라이언트	클라이언트 1 서버 4 서버 5

도면12

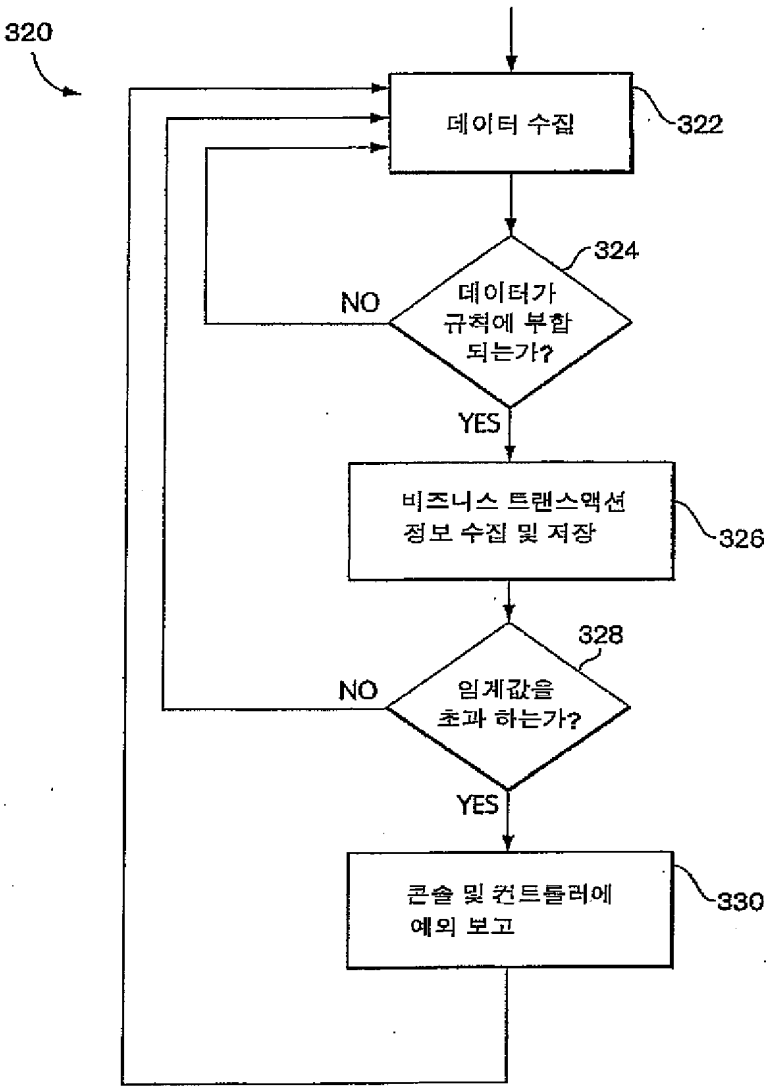
280 282 284 286

데이터 수집 ID	명세	캐시값, 구성요소
A	클라이언트 CPU 시간	3초, 클라이언트 1
A	클라이언트 CPU 시간	2초, 클라이언트 2
B	서비스 되는 요청의 수	5, 서버 1
B	서비스 되는 요청의 수	34, 서버 2
⋮	⋮	⋮

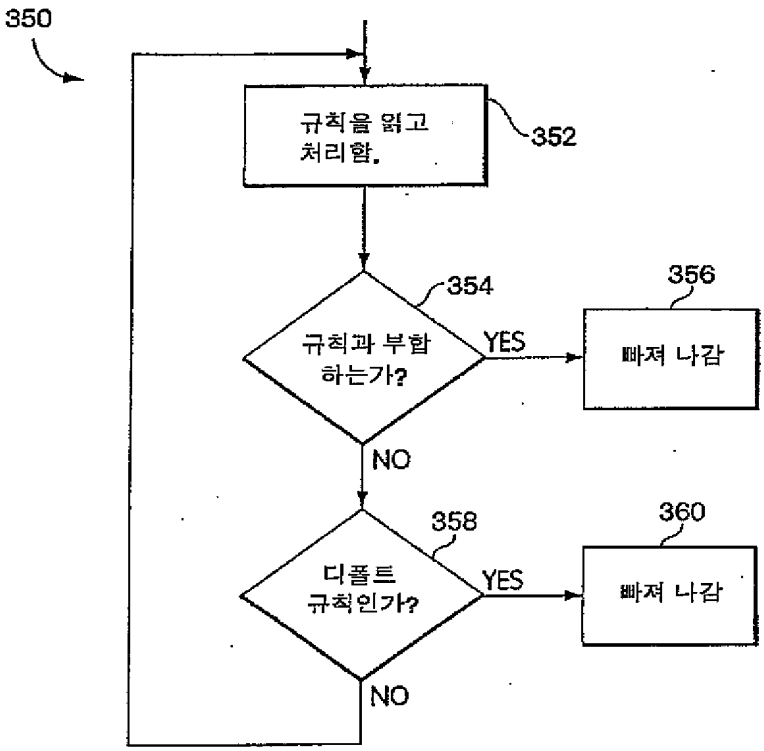
도면13



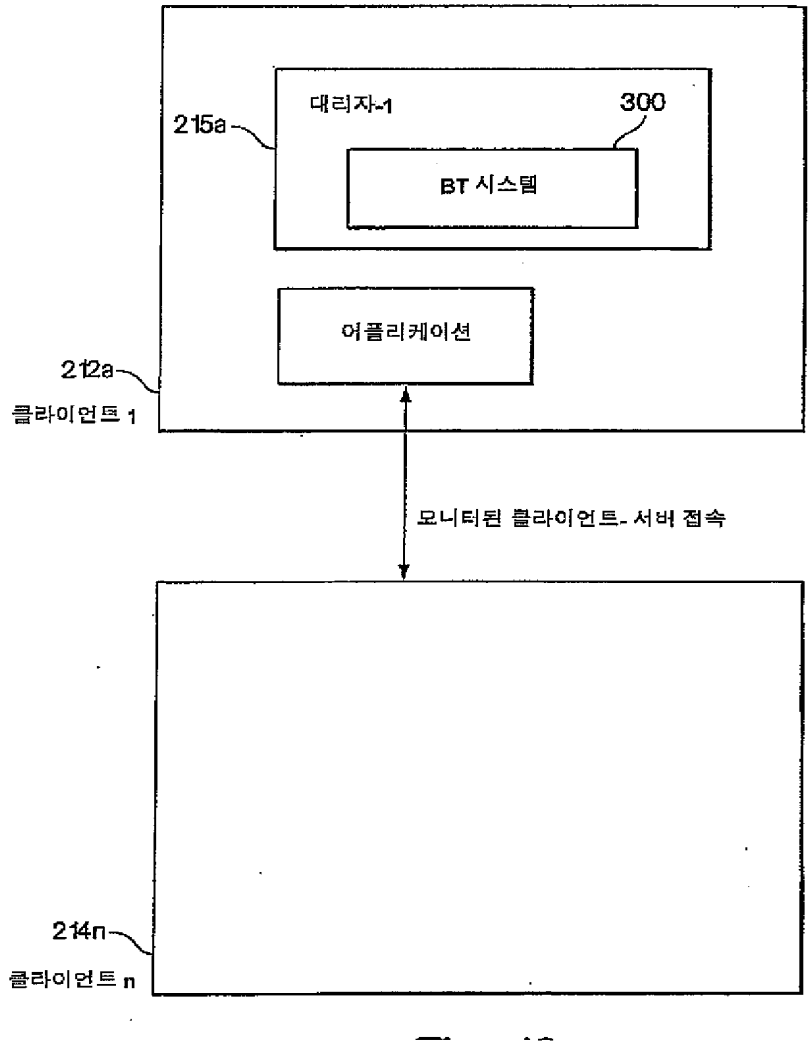
도면14


320
데이터 수집
322
324
데이터가 규칙에 부합 되는가?
NO
YES
비즈니스 트랜스액션 정보 수집 및 저장
326
328
임계값을 초과 하는가?
NO
YES
콘솔 및 컨트롤러에 예외 보고
330

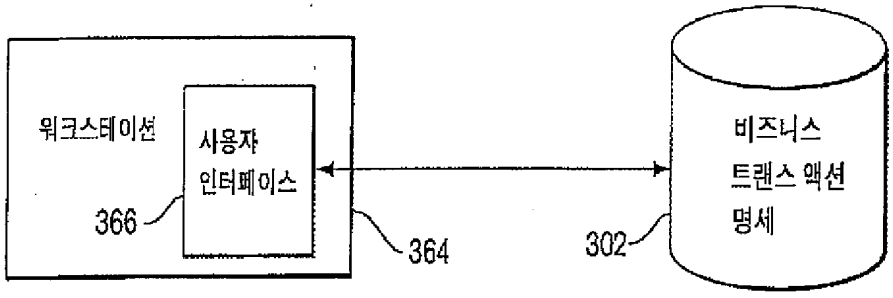
도면15


350
규칙을 읽고
처리함.
352
354
규칙과 부합
하는가?
YES
356
빠져 나감
NO
358
디폴트
규칙인가?
YES
360
빠져 나감
NO

도면16


대리자-1
300
215a
BT 시스템
어플리케이션
212a
클라이언트 1
모니터된 클라이언트-서버 접속
214n
클라이언트 n

도면17


워크스테이션
사용자
인터페이스
366
364
302
비즈니스
트랜스 액션
명세